Stable Feature Extraction in Aerial Reconnaissance Images

Final Technical Report

by

Darrel L. Chenoweth

June 1998

University of Louisville Research Foundation, Inc.

Louisville Kentucky

Grant Number N00014–92–J–4096

Principal Investigator: Darrel L. Chenoweth

REPORT DOCUMENTATION PAGE

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data source: gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of thi collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferso. Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE June 1998	3. REPORT TYPE AND DATES COVERED Final Report 15 August 92–12 August 96

4. TITLE AND SUBTITLE Stable Feature Extraxtion in Aerial Reconnaissance Images	5. FUNDING NUMBERS Grant – N00014-92-4069

6. AUTHOR(S)

Darrel L. Chenoweth

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University of Louisville, Research Foundation Louisville, Kentucky 40292	8. PERFORMING ORGANIZATION REPORT NUMBER EEDEPSCOR-01

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research (Attn: Mr. William J. Miceli) Program Officer (RF Surveillance Technology) 800 N. Quincy Street Arlington, VA 22217-5660	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT Unlimited distribution	12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The problem of extracting stable features in aerial reconnaissance images is addressed in this report. Fractal models of image features are proposed and the concept of fractal error is introduced. Fractal error is a metric that quantifies the "closeness of fit" of the image feature to a fractal model. Using this metric one can discriminate between natural and man-made features in an image. Fractal error also presents a useful approach to extracting edges in images. Methods are given for computing and approximating fractal error using networks and genetic algorithms.

14. SUBJECT TERMS Pattern recognition, fractals, image segmentation, mission, planning			15. NUMBER OF PAGES
			16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

TOTAL P.03

PLEASE CHECK THE APPROPRIATE BLOCK BELOW:

-AO #_____

☐ __2__ copies are being forwarded. Indicate whether Statement A, B, C, D, E, F, or X applies.

☒ DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE: DISTRIBUTION IS UNLIMITED

☐ DISTRIBUTION STATEMENT B:
DISTRIBUTION AUTHORIZED TO U.S. GOVERNMENT AGENCIES
ONLY; (Indicate Reason and Date).. OTHER REQUESTS FOR THIS
DOCUMENT SHALL BE REFERRED TO (Indicate Controlling DoD Office).

☐ DISTRIBUTION STATEMENT C:
DISTRIBUTION AUTHORIZED TO U.S. GOVERNMENT AGENCIES AND
THEIR CONTRACTORS; (Indicate Reason and Date). OTHER REQUESTS
FOR THIS DOCUMENT SHALL BE REFERRED TO (Indicate Controlling DoD Office).

☐ DISTRIBUTION STATEMENT D:
DISTRIBUTION AUTHORIZED TO DoD AND U.S. DoD CONTRACTORS
ONLY; (Indicate Reason and Date). OTHER REQUESTS SHALL BE REFERRED TO
(Indicate Controlling DoD Office).

☐ DISTRIBUTION STATEMENT E:
DISTRIBUTION AUTHORIZED TO DoD COMPONENTS ONLY; (Indicate
Reason and Date). OTHER REQUESTS SHALL BE REFERRED TO (Indicate Controlling DoD Office).

☐ DISTRIBUTION STATEMENT F:
FURTHER DISSEMINATION ONLY AS DIRECTED BY (Indicate Controlling DoD Office and Date) or HIGHER
DoD AUTHORITY.

☐ DISTRIBUTION STATEMENT X:
DISTRIBUTION AUTHORIZED TO U.S. GOVERNMENT AGENCIES
AND PRIVATE INDIVIDUALS OR ENTERPRISES ELIGIBLE TO OBTAIN EXPORT-CONTROLLED
TECHNICAL DATA IN ACCORDANCE WITH DoD DIRECTIVE 5230.25, WITHHOLDING OF
UNCLASSIFIED TECHNICAL DATA FROM PUBLIC DISCLOSURE, 6 Nov 1984 (Indicate date of determination).
CONTROLLING DoD OFFICE IS (Indicate Controlling DoD Office).

This document was previously forwarded to DTIC on _____ (date) and the
AD number is _____.

In accordance with provisions of DoD instructions, the document requested is not supplied because:

It will be published at a later date. (Enter approximate date, if known).

Other. (Give Reason)

D Directive 5230.24, "Distribution Statements on Technical Documents," 18 Mar 87, contains seven distribution statements, as
cribed briefly above. Technical Documents must be assigned distribution statements.

DARREL L. CHENOWETH
Print or Type Name

(502) 852-7948
Telephone Number

David J. Chenoweth
Authorized Signature/Date

Contents

Chapter 1

Abstract

In 1990, the Naval Avionics Center/Indianapolis (NAC) contracted with the University of Louisville (UofL) to perform research and development of image processing algorithms to support the Digital Scene Matching Area Correlator (DSMAC) mission planning function. DSMAC is the map matching navigation system used by the Tomahawk Land Attack Missile for terminal navigation updates. Subsequently, DEPSCOR Grant N00014-92-J-4096 was awarded to the University of Louisville by ONR to support doctoral student research related to the DSMAC mission planning research. This report constitutes the final technical report for grant N00014-92-J-4096. The grant period was three years, with a one-year no-cost extension.

One of the major concerns in mission planning for DSMAC is the selection of ground scenes that contain an adequate number of stable features, i.e., scene features that remain constant regardless of diurnal and seasonal variations. This is important because it maximizes the duration of the time window in which the mission planning reference maps can be used. Gray scale reconnaissance imagery is used to create reference maps, and the research to date by UofL has concentrated on the development of image segmentation algorithms and software for this purpose. We have developed a gray scale/texture segmenter for segmenting reconnaissance images.

The UofL gray scale/texture segmenter utilizes low level gray scale and texture features, together with a Bayesian statistical classifier, to classify image pixels into a set of image feature classes, one of the classes being the *unknown* class. One of the problems is the unavoidable misclassification of image features. We developed a rule-based high level segmentation algorithm that will interface with the existing low level segmenter and accomplish an important function: it systematically applies heuristics to correct misclassified regions. The knowledge base requires the extraction of region attributes, and a number of transforms appear to be promising for this purpose. Fractal dimension and fractal error are useful attributes that describe texture and *fractalness*. For example, cultural regions have high fractal error (they typically do not fit fractal models), and they are generally stable relative to seasonal and diurnal cycles. We have developed techniques for efficiently and accurately extracting fractal error and fractal dimension. An analysis of fractal error for cultural object detection in aerial images, including synthetic aperture radar (SAR) is presented. Because fractal error can be computationally expensive, approximations to the measure were developed.

The chord transform appears to be a useful metric for determining the cultural and natural feature content in an aerial image. This was developed and tested, providing a robust geometric feature for segmentation in aerial imagery.

Chapter 2

Overview

2.1 Introduction

This report is laid out in four distinct chapters. Chapter 1 is an abstract describing the problem statement and the research accomplished under ONR Grant N00014-92-J-4096. Chapter 2 outlines the internal and external publications supported by this grant. The fractal error metric and the chord transform play an important role in the stable feature extraction and segmentation of aerial reconnaissance images documented in two Ph.D. dissertations, two M.Eng. thesis and ten journal and conference publications resulting from this grant. Thus, the fractal error metric and the chord transform are described in detail in Chapters 3 and 4, respectively. The appendix contains two sections. The first section contains the unabridged text of each of the Ph.D. dissertations. The second section contains the ten journal and conference publications.

2.2 Internal Publications

ONR Grant N00014-92-J-4096 has supported, either directly or indirectly, two Ph.D. dissertations and two M.Eng. thesis. These are listed below:

- B. E. Cooper, *Fractional Brownian Motion for Representing Natural Image Texture*, Ph.D. dissertation, University of Louisville, 1994.

- E. D. Jansing *Feature Extraction and Edge Detection Using a Fractal-Based Metric*, Ph.D. dissertation, University of Louisville, 1997.

- O. Haidar, *A Knowledge-Based System for Improving Gray Shade/Texture Segmentation*, M.Eng. thesis, University of Louisville, 1992.

- B. S. Allen, *Fractal Error Approximation Using Neural Networks*, M.Eng. thesis, University of Louisville, 1997.

Cooper [1] examined the possibility of using fractional Brownian motion (fBm) models for describing texture. Prior research demonstrated that fBm may be used to model and

4

subsequently characterize visual textures in aerial imagery. He argues, however, that much of the prior work has focused on fractal dimension, lacunarity or variants of fractal dimension, while little work has examined the relationship between the actual image data and the fractal model. Cooper thus proposed a measured called *fractal error* and introduced this measure as a texture feature. Cooper used feature extraction in aerial imagery as a basis for his analysis. A localized version of fractal error was developed, which Cooper found to be a robust feature for segmentation when compared to other features, such as the busyness operator.

Jansing [2] investigated fractal error as a feature in aerial image segmentation and industrial edge detection. Jansing found fractal error to be useful in several different aerial images, including synthetic aperture radar (SAR). Applying this method to industrial images, he developed a new edge detection technique that proved to be robust, even in the presence of noise. In the course of the investigation, Jansing encountered the dilemma that calculating fractal error can be computationally expensive. This prompted the implementation of an approximation of fractal error using genetic algorithms. While Cooper [1] developed a Bayesian segmentation method, Jansing designed a novel 2-D entropic segmentation algorithm. This algorithm was shown to be accurate for cultural object detection in aerial images, using gray shade and fractal error as features.

The knowledge-based system for improving gray-shade/texture segmentation was designed and implemented by Haidar [3]. This system uses selected attributes from selected segments in a segmented image. The selected attributes are used by heuristic rules in the knowledge-base to correct misclassifications. By using a set of ten attributes, including geometric information of the segments provided by the chord transform, and four rules, Haidar found his system corrected most of the misclassification in a segmented image.

While Jansing [2] developed his approximation of fractal error using genetic algorithms, Allen [4] also recognized and quantified the computational expense of fractal error. Allen proposed using an artificial neural network as a universal approximator. The neural network architectures used were a multilayer feedforward network (MLFN) and a functional link network. While the functional link network was shown to be much faster than the MLFN, the MLFN gave a very accurate approximation. After tuning the network's parameters, Allen demonstrated that a five neuron hidden layer provided an accurate and fast approximation to fractal error. This network was independent of the input images, allowing for hardware implementation in the future.

2.3 External Publications

The following journal and conference publications have been directly or indirectly supported by this ONR grant:

- B. E. Cooper, D. L. Chenoweth and J. E. Selvage, "Fractal Error for Detecting Man-Made Features in Aerial Images," *Electronics Letters*, Vol. 30, No. 7, pp. 554-555, 1994.

- D. L. Chenoweth, B. E. Cooper and J. E. Selvage, "Aerial Image Analysis using Fractal-Based Models," *Proceedings of the IEEE Aerospace Conference*, Vol. 2, pp. 277-285, Snowmass, CO, Feb 1995.

- V. E. Gold, D. L. Chenoweth and J. E. Selvage, "Image Segmentation using Modified Neural Network Techniques," *Proceedings of the 1196 IS& T/SPIE Symposium on Electronic Imaging: Science & Technology*, San Jose, CA, Jan/Feb 1996.

- J. E. Selvage, D. L. Chenoweth and V. E. Gold, "Geometric Feature Extraction using the Chord Transform," *Proceedings of the IEEE Aerospace Conference*, Vol. 2, Snowmass, CO, Feb 1996.

- E. D. Jansing, D. L. Chenoweth and J. Knecht, "Detection of Man-Made Objects in Synthetic Aperture Radar (SAR) Images using Fractal Error," *Proceedings of the IEEE Aerospace Conference*, Vol. 1, Snowmass, CO, Feb 1997.

- E. D. Jansing, B. S. Allen and D. L. Chenoweth, "Edge Enhancement using the Fractal Error Metric," *Proceedings of the 1st International Conference on Engineering Design and Automation*, Bangkok, Thailand, Mar 1997.

- B. S. Allen, E. D. Jansing and D. L. Chenoweth, "Neural and Genetic Approximations to Fractal Error," *Proceedings of the IEEE Aerospace Conference*, Snowmass, CO, Mar 1998.

- B. E. Cooper, E. D. Jansing, D. L. Chenoweth and J. E. Selvage, "Fractal Error For Image Analysis," in revision for *Pattern Recognition*.

- E. D. Jansing, T. A. Albert and D. L. Chenoweth, "Two-Dimensional Entropic Segmentation Of Cultural Objects In Aerial Imagery," conditionally accepted for publication in *Pattern Recognition Letters*.

- E. D. Jansing and D. L. Chenoweth, "A Genetic Approximation To Fractal Error," submitted for review to *Evolutionary Optimization*.

Chapter 3

Fractal Error

3.1 Introduction

Fractal error is an image processing metric that can be used to locate man-made features in aerial images. The metric can aid photointerpreters in locating targets in aerial reconnaissance images. Fractal error was developed for this purpose by Cooper et al. [1], [5], [6], [7]. Since its development, Jansing et al. [2], [8] have shown that the fractal error metric also works well for extracting features in synthetic aperture radar (SAR) images. Jansing et al. [2], [9] have also shown that the fractal error metric can be used for locating edge pixels in industrial images. The fractal error metric has a wide range of applications; however, some applications require real-time analysis. The main disadvantage of the fractal error algorithm is that it can take several seconds to compute on large images. Jansing et al. [2], [10] have developed a genetic approximation of fractal error. Allen et al. [4], [11] also propose an approximation of fractal error using artificial neural networks. These approximations give usable representations of fractal error, while providing real-time or near real-time performance. The neural network provides an accurate representation of fractal error, while the genetic algorithm does in fact preserve all of the desired features of the original fractal error image. The genetic algorithm, on the whole, has been shown to be computationally faster than the neural network.

This fractal error metric has gained acceptance in the literature. For example, Lin et al. [12] use fractal error to detect targets in X-band and millimeter wave (MMW) radar images. They proposed some modest modifications to the metric to improve its performance with MMW radar images.

3.2 Definition Of A Fractal

It is well known that many textures and scenes can be modeled as *fractals*. A fractal, according to La Brecque [13], "has a rough shape to one degree or another made of parts which, when magnified, resemble the whole." It is also well known that literature describing fractals often lacks precision when attempting to define what a fractal is. However, the reader of fractal geometry and theory can turn to Falconer [14] for a detailed description of the properties of fractals.

Definition 1 *The set F is a fractal, if it has the following properties:*

1. *F has a fine structure; that is, detail on arbitrarily small scales.*

2. *F is too irregular to be described in traditional geometrical language, both locally and globally.*

3. *F often has some form of self-similarity, perhaps approximate or statistical.*

4. *Usually, the "fractal dimension" of F (defined in some way, and there are several unique definitions) is greater than its topological dimension.*

5. *In most cases of interest, F is defined in a very simple way, perhaps recursively (e.g., the Julia or Mandelbrot Sets).*

3.3 Fractal Dimension

How are fractals distinguished between one another? How does one measure the size of fractals? What measure can be used to compare and contrast fractals?

Fractal dimension is the measure that is generally used to distinguish between fractals, giving the fractals a measurement of "size". This numeric representation attempts to quantify a subjective quality which one might have about how densely the fractal occupies the space in which it exists.

Fractal dimension is just as difficult to define as fractals themselves. Mandelbrot [15], Falconer [14], Peitgen et al. [16] and Edgar [17] provide excellent discussions of many different fractal dimension definitions. Each fractal dimension definition has a distinct style. Although the definitions are all related, Peitgen [16] claims that some definitions make sense in certain cases, while other definitions may not be appropriate in the same case. Experience and heuristics prompt the selection of an appropriate fractal dimension definition, according to the application.

3.4 Lacunarity

Mandelbrot [15] defined another measure for fractals. *Lacunarity* describes the "holiness" ([18], pg. 236) of an occupied fractal lattice. The origin of the name lacunarity can be appreciated by looking at an image of cork in Figure 3.1. This image is part of a collection of texture images, presented by Brodatz [19]. From the Latin word "lacona," which means gap, lacunarity represents the gaps within a fractal structure. Thus, the percentage of spaces between the cork in Figure 3.1 is the measure of lacunarity. Practically, lakes or other natural objects within aerial images may be classified by using lacunarity as a feature measure.

3.5 Fractional Brownian Motion

Fractals may occur in many different forms. Mandelbrot [20] was the first to define *fractional Brownian motion (fBm)*. Brownian motion refers to the erratic motion of small suspended
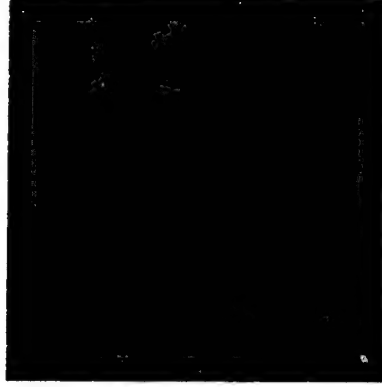
8

Figure 3.1: Texture image of cork.

particles, resulting from random collisions with other particles. Fractional Brownian motion is an extension of this model. Cooper [1] gives an excellent description of fBm; it will be summarized here.

The function of fBm is defined as the differences between successive samples. Let $B_H(t)$ represent a fBm signal, where t is a vector containing E independent variables. Then the increment of the fBm signal is described as $\Delta B_H = B_H(t_2) - B_H(t_1)$, where t_1 and t_2 are two distinct points in time. The measure ΔB_H is normally distributed with a mean of zero and a variance proportional to the $2H$ power of the Euclidean distance. The mean takes the form of

$$E[B_H(t_2) - B_H(t_1)] = 0. \tag{3.1}$$

Likewise, the variance is defined as

$$\text{Var}[B_H(t_2) - B_H(t_1)] = \sigma^2 |t_2 - t_1|^{2H}. \tag{3.2}$$

where σ^2 is the proportionality constant of the variance and H is the Hurst parameter, which must be strictly $0 < H < 1$. Note that when $H = 0.5$, the fractional Brownian motion model is equivalent to the classical Brownian motion model. The value of H can be used to describe the fractal dimension D as

$$D = E + 1 - H, \tag{3.3}$$

where E is the Euclidean dimension (or, the number of independent variables of t). It is therefore easy to see that small values of H produce high fractal dimension and large values of H produce a low fractal dimension. Combining the equations for mean and variance, while also taking into account the fractal dimension, we can arrive at the following relationship:

$$E[|B_H(t_2) - B_H(t_1)|] = k|t_2 - t_1|^H. \tag{3.4}$$

The above equation is the fundamental basis for the fractal error metric. The regions observed in an actual image may be a combination of many different textures. Each texture can be represented by its fractal dimension. However, before attempting to determine the fractal dimension, it is useful to know how well a region (or window) may fit the fractal model. Thus, measuring the error produced when estimating the fractal dimension will give

9

a useful metric in order to determine the "fractalness" of a region in the image. A small error will indicate that a region fits the fractal model well, and thus can be considered fractal. Conversely, a large error will indicate that the region fits poorly into the fractal model and thus is probably not fractal and the fractal dimension measure is useless. Mathematically, this can be defined as

$$E[\|G[\mathbf{x_2}] - G[\mathbf{x_1}]\|] = k|\mathbf{x_2} - \mathbf{x_1}|^H \tag{3.5}$$
$$E[|\Delta G_{|\Delta\mathbf{x}|}|] = k|\Delta\mathbf{x}|^H \tag{3.6}$$

where G is the region or window in the image and $\mathbf{x}$ is the measured distances within the region. Estimates of H and k can be found by using a linear regression scheme,

$$\ln E[|\Delta G_{|\Delta\mathbf{x}|}|] = \ln k + H \ln |\Delta\mathbf{x}|. \tag{3.7}$$

These estimates, $\overline{H}$ and $\overline{k}$ can then be used to calculate the error with the following equation:

$$error_{|\Delta\mathbf{x}|} = E[|\Delta G_{|\Delta\mathbf{x}|}|] - \overline{k}|\Delta\mathbf{x}|^{\overline{H}}. \tag{3.8}$$

Using a "center-oriented" window (i.e., a square window of NxN, where N is strictly odd), there will be five, nine, or fourteen error values, given the window is 5x5, 7x7, or 9x9 respectively. Thus, a cumulative error for the model can be given by the root mean square error

$$RMS\ Error = \sqrt{\frac{1}{n} \sum_{|\Delta\mathbf{x}|} (error_{|\Delta\mathbf{x}|})^2}. \tag{3.9}$$

Thus, using the RMS error, it is easily determined whether or not a pixel with a surrounding 5x5, 7x7 or 9x9 region is fractal in nature.

3.6 Fractal Error Algorithm

Using the method described in the previous section, Cooper developed an algorithm to calculate the fractal error for each pixel in a scene. This algorithm is described in detail in Table 3.1.

The following will outline the steps of the entire fractal error algorithm. Note that this example will produce a single number that will represent the error for the center pixel in relation to its neighbors. Figure 3.2 represents a sample 5x5 window. Since the localized neighborhood is 5x5, there are five unique distances in the window, as shown in Figure 3.3.

Table 3.2 represents the absolute values of the differences of the gray scales over the unique sets of distances in the neighborhood. Thus, the gray scale value of each pixel that has a distance of 1.41 is subtracted from the gray scale value of the center pixel. Their absolute values are averaged to give $E[\|\Delta G\|]$ for each unique set of distances.

Using Equation 3.7, we can obtain estimates for the Hurst parameter and the proportionality constant, H and k respectively. These estimates can be found using linear regression [21]. If the linear model takes the form

$$y = \beta_0 + \beta_1 x, \tag{3.10}$$

Table 3.1: Fractal Error Algorithm

1. Define a 5x5, 7x7, or 9x9 sliding window.
2. Calculate $\Delta\mathbf{x}$ and $E[|\Delta G|]$ for each pixel in the neighborhood of the sliding window.
3. Using linear regression, find the slope and the y-intercept for each unique $\Delta\mathbf{x}$ in the window from the equation $\ln(E[|\Delta G_{|\Delta x_i|}|]) = \ln(k) + H|\Delta x_i|$.
4. Derive $\overline{H} = \text{slope}$ and $\overline{k} = \exp(\text{y-intercept})$ from the above relationship.
5. Using $error_{|\Delta x_i|} = E[|\Delta G_{|\Delta x_i|}|] - \overline{k}|\Delta x_i|^{\overline{H}}$, calculate the fractal error for each unique $\Delta\mathbf{x}$.
6. Compute RMS error by $RMS\ Error = \sqrt{\frac{1}{n}\sum_{|\Delta\mathbf{x}_i|}(error_{|\Delta\mathbf{x}_i|})^2}$.
7. Save RMS error for that pixel, move the window, and repeat the process over the entire scene.

250	200	220	200	200
175	210	170	159	100
110	100	120	115	100
96	200	205	210	211
95	201	197	205	200

Figure 3.2: A sample 5x5 window.

2.83	2.23	2.00	2.23	2.83
2.23	1.41	1.00	1.41	2.23
2.00	1.00		1.00	2.00
2.23	1.41	1.00	1.41	2.23
2.83	2.23	2.00	2.23	2.83

Figure 3.3: The Euclidean distances over the 5x5 neighborhood.

Table 3.2: Distances From The Center Pixel (Δx) And The Expected Value Of The Absolute Difference In Gray Scale Relevant To The Center Pixel ($\mathrm{E}[|\Delta G|]$)

Δx	$\ln \Delta x$	$\mathrm{E}[|\Delta G|]$	$\ln \mathrm{E}[|\Delta G|]$
1.000	0.000	40.00	3.69
1.414	0.347	74.75	4.31
2.000	0.693	51.75	3.95
2.236	0.805	91.50	4.51
2.828	1.040	78.75	4.37

Table 3.3: Calculated Error From Fractional Brownian Motion Model

Δx	error
1.00	-6.06
1.41	18.4
2.00	-17.3
2.24	17.7
2.83	-5.9

then estimates for β_1 (slope) and β_0 (y-intercept) are defined as

$$\widehat{\beta_1} = \frac{\Sigma(x_i - \overline{x})y_i}{\Sigma(x_i - \overline{x})^2} \tag{3.11}$$

$$\widehat{\beta_0} = \overline{y} - \widehat{\beta_1}\overline{x}, \tag{3.12}$$

where $\overline{x}$ is the sample mean of x and $\overline{y}$ is the sample mean of y. In our particular case, x represents the "$\ln \Delta x$" term in Equation 3.7 and y represents the "$\ln \mathrm{E}[|\Delta G|]$" term in the same equation. It easy to see that $\overline{x} = 0.577$ and $\overline{y} = 4.17$. The estimates of β_0 and β_1 can thus be calculated,

$$\widehat{\beta_1} = \frac{\sum_{i=0}^{5}(x_i - 0.577)y_i}{\sum_{i=0}^{5}(x_i - 0.577)^2} \tag{3.13}$$

$$= 0.585 \tag{3.14}$$

$$\widehat{\beta_0} = 4.17 - 0.577\widehat{\beta_1} = 3.83$$

The Hurst parameter estimate, $\overline{H}$, is equivalent to the slope of the linear model, that is, $\overline{H} = \widehat{\beta_1}$. The proportionality constant estimate, $\overline{k}$, is equivalent to the y-intercept of the linear model, $\overline{k} = \exp(\widehat{\beta_0})$. Therefore, $\overline{H} = 0.585$ and $\overline{k} = 46.1$.

Errors with respect to each unique distance set can be calculated with Equation 3.8. Table 3.3 shows the result of using Equation 3.8 in this example.

The overall RMS error, as defined by Equation 3.9, is 14.31. This number represents the "fractalness" of the center pixel relative to its neighbors. Decisions regarding the fractalness

of the pixel are typically made in reference to the entire image. Thus, if the range of fractal errors is from 0 to 15, this pixel is not likely to be fractal in nature.

3.7 Results

Shown in Figure 3.4 is a small image of Alameda, CA. This is a high-contrast image, containing both natural and cultural features. Figure 3.4 shows the gray shade image, a contour map of the image and its histogram.
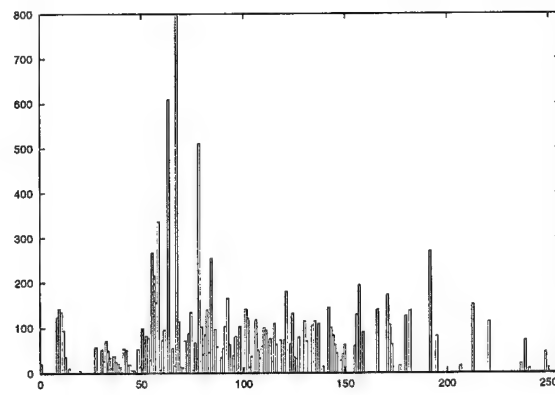
Figure 3.5 shows the resulting fractal error measure for Figure 3.4. The figure again shows the normalized gray shade image, contour map and histogram of the fractal error result. It is not difficult to see that those areas with a fractal error higher than 5 represent cultural objects (refer to the contour map). Likewise, the areas that have light gray to white pixels in the normalized gray shade image represent cultural objects, while the darker pixels represent natural terrain.

(a) Gray Shade Image

250 -------
200 -------
150 -------
100 -------
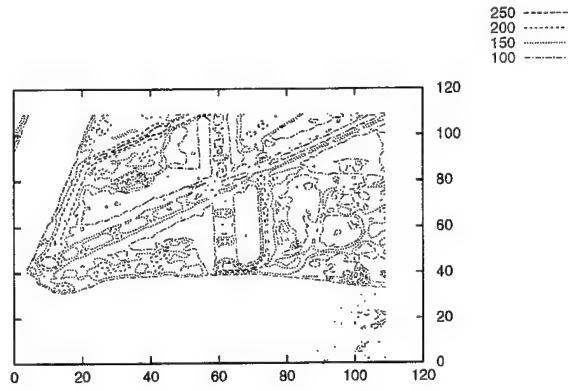
(b) Contour Map

(c) Histogram

Figure 3.4: Aerial Image of Alameda, CA.

14

(a) Normalized Gray Shade Image
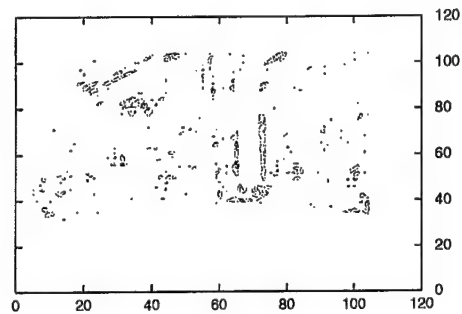
(b) Contour Map
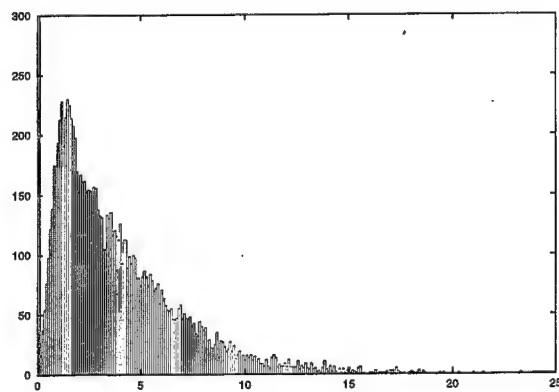
(c) Histogram

Figure 3.5: Resulting Fractal Error Image of Alameda, CA.

Chapter 4

Chord Transform

4.1 Introduction

In classification systems, it is sometimes necessary to know whether a segment in a segmented image is a natural segment or cultural segment. Thus, it may be useful to include geometric information as a feature in segmentation. The chord transform reveals information about the geometric structure of an image segment, and it can be used to extract a limited set of geometric attributes of an image segment. Haidar [3] and Selvage [22] give a detailed description of the chord transform. It will be summarized here. Results are presented and discussed in [3] and [22].

4.2 The Chord Transform

In order to perform the chord transform on an image segment, it is necessary to preprocess the image to obtain the edge pixels of the segment. The general idea can be shown with the assistance of Figure 4.1.

The gradient vectors, G_1 and G_2 in Figure 4.1, are found for each the edge pixels of the segment. The transform starts by choosing one edge pixel and performing the chord
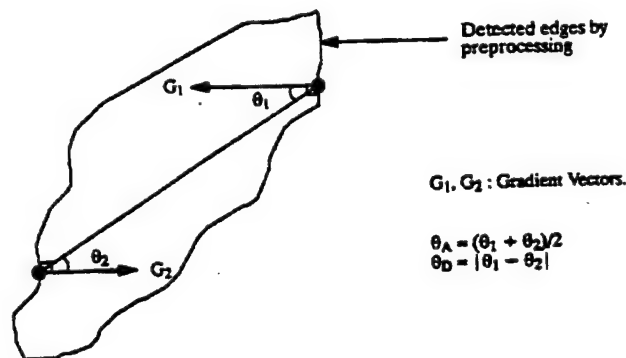
Detected edges by preprocessing

G_1, G_2 : Gradient Vectors.

$\theta_A = (\theta_1 + \theta_2)/2$
$\theta_D = |\theta_1 - \theta_2|$

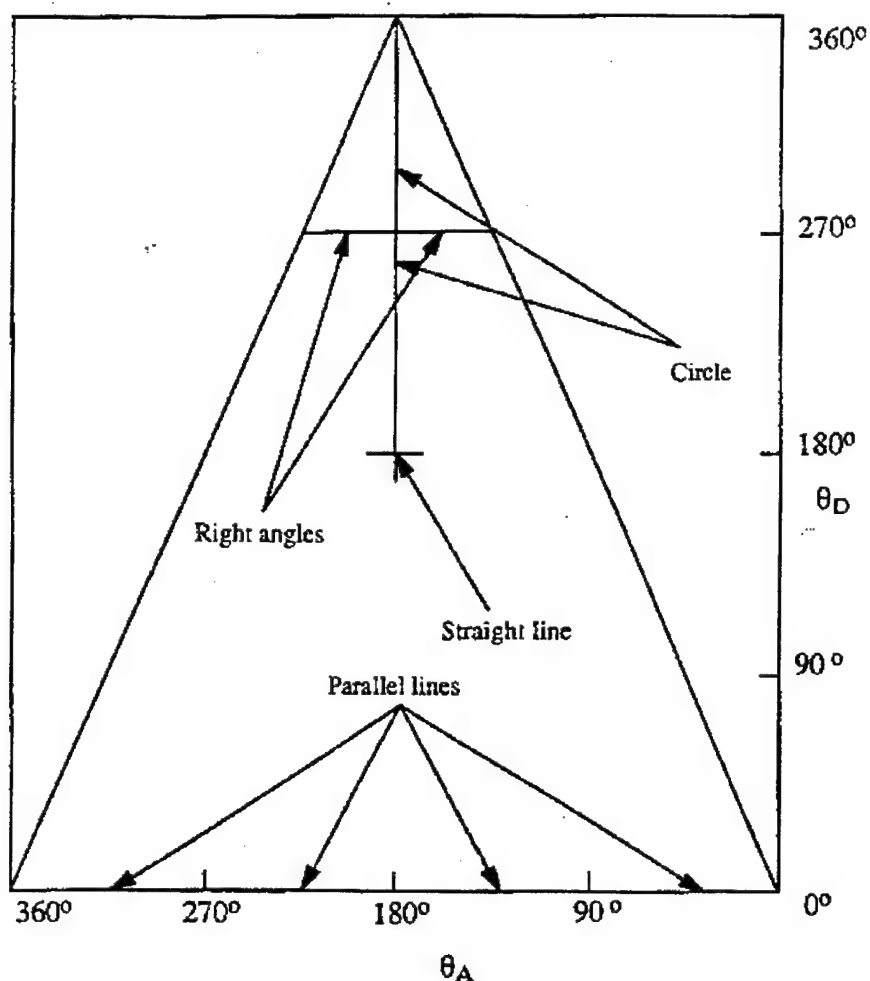Figure 4.1: The general case of the chord transform.

16

Figure 4.2: The quantized angle histogram and the signatures of certain geometric attributes.

procedure for that edge pixel all all other possible edge pixels of the segment. By measuring the counterclockwise angle between both G_1 and G_2 for any two given edge pixels and their connecting vector (called a chord), the measurement of θ_1 and θ_2 can be found. The two angles define coordinates of a *quantized angle histogram*, shown in Figure 4.2, with features θ_A and θ_D. Each possible pair of edge pixels is taken into account for the transform.

The chord transform isolates identifying characteristic "signatures" for straight lines, perpendicular lines, parallel lines, and circles in an image. Figure 4.2 illustrates the areas in the histogram for certain signatures of the geometric attributes. For example, the signature of a straight line is represented by a single point at coordinates $(180°, 180°)$. Because the chord transform relies on geometric information between edge pixel pairs, the transform is rotation and scale invariant.

Table 4.1: The chord transform algorithm.

1.	Get segment edge point list.
2.	Compute gradient along segment edge.
3.	Get pair of edge points.
4.	Compute θ_1 and θ_2.
5.	Compute θ_A and θ_D.
6.	Increment histogram pair (θ_A, θ_D).
7.	All pairs done? Yes–Stop, No–Go to step 3.

4.3 The Algorithm

Table 4.1 describes the steps that are needed to perform the chord transform of a particular image segment. The first goal in finding the chord transform is to extract from a given segment a list of coordinates that comprise the edge of the segment. The edge map for a particular segment should be one pixel wide and closed. The first pixel is found by searching the image in the x (or row) direction until an edge pixel is found. Successive pixels are found by a clockwise search around a 3x3 neighborhood.

Once the edge map and first edge pixel are located, then the gradients for each edge pixel is found, relative to its neighboring edge pixels. These gradients are used, along with the line segment connecting any two edge pixels in the map, to compute the angular information. This is accomplished by taking two points at a time from the edge map. Since there are N_e pixels, taken two at a time, there will be $N_t = N_e(N_e - 1)/2$ combinations of unique pairs of edge pixels.

Let any two points from the edge map be X_1 and X_2. A line segment joins the two points. This line (or chord) is the reference direction for angle measurements, as illustrated in Figure 4.3. Let w be the angular distance from the reference direction to the chord, measured counterclockwise. A similar definition is made for X_2 and w_p. Then w and w_p are used to calculate θ_1 and θ_2:

$$
\begin{aligned}
J &= J_2 - J_1 \\
w &= \arctan(J/I) \\
\theta_1 &= G_1 - w,
\end{aligned}
\tag{4.1}
$$

$$
\begin{aligned}
I &= I_2 - I_1 \\
w_p &= w + (\pi/2) \\
\theta 2 &= G_2 - w_p,
\end{aligned}
\tag{4.2}
$$

where J_1, J_2, I_1, I_2 are the row and column coordinates for the two edge pixels. Once θ_1 and θ_2 is calculated for each pair of edge pixels, they are used to find θ_A and θ_D which represent
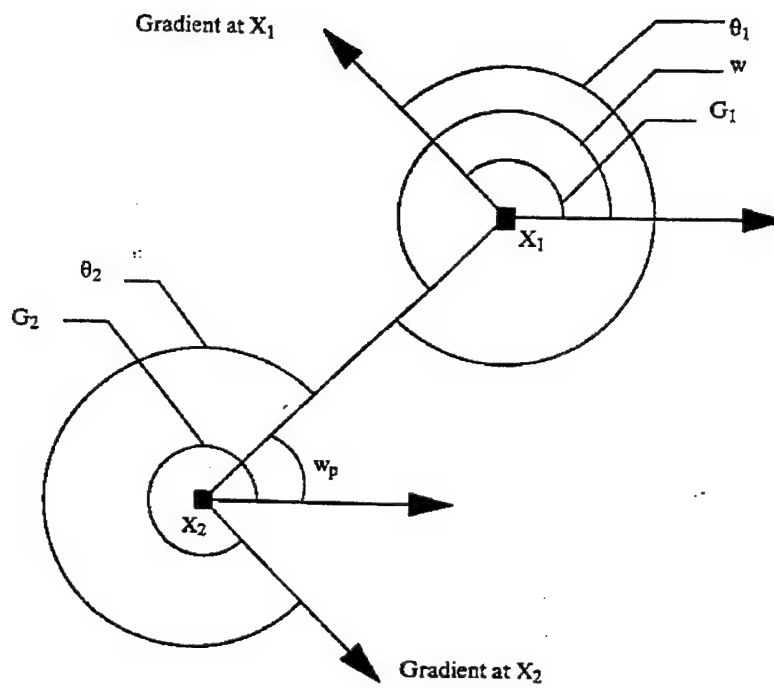
18

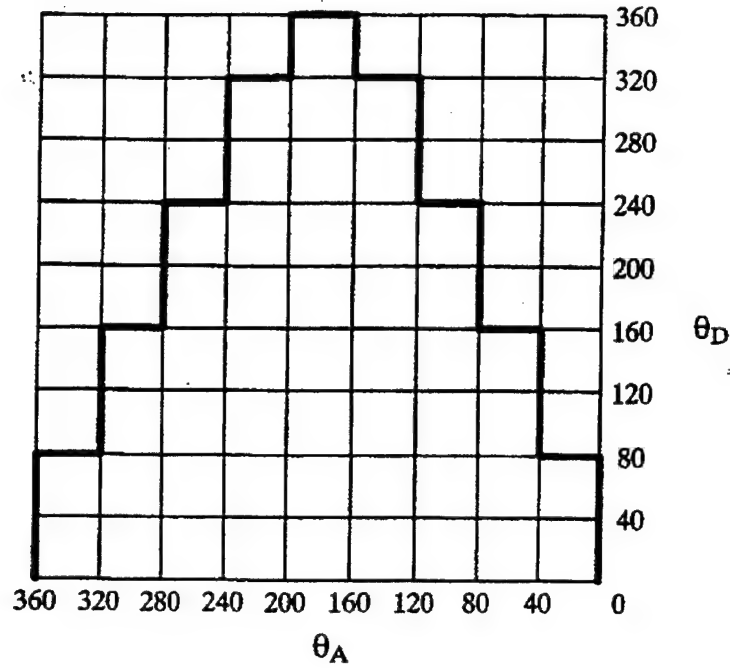Figure 4.3: The definition of θ_1 and θ_2.

Figure 4.4: The selected attributes from the quantized angle histogram.

the average and the difference, respectively,

$$\theta_A = \frac{(\theta_1 + \theta_2)}{2} \tag{4.3}$$

$$\theta_D = |\theta_1 - \theta_2| \tag{4.4}$$

Once all N_t angle pairs are found, they are compiled into the quantized angle histogram. The histogram measures the frequency of occurrence of each possible pair of angle. This histogram can then be sliced into sections to be used for feature extraction. Figure 4.4 shows a possible sectioning of the histogram. It can be shown that the elements outside the bold "triangle" always have a value of zero; therefore, only the values inside this triangle provide useful information. These blocks inside the triangle can be used to make a 49 element feature vector, describing the geometric structure of the image segment.

20

Bibliography

[1] B. E. Cooper, *Fractional Brownian Motion For Representation Of Natural Image Texture*. PhD thesis, University of Louisville, 1994.

[2] E. D. Jansing, *Feature Extraction and Edge Detection Usning A Fractal-Based Metric*. PhD thesis, University of Louisville, 1997.

[3] O. Haidar, "A knowledge-based system for improving gray shade/texture segmentation," Master's thesis, University of Louisville, 1992.

[4] B. Allen, "Fractal error approximation using neural networks," Master's thesis, University of Louisville, 1997.

[5] B. E. Cooper, D. L. Chenoweth, and J. E. Selvage, "Fractal error for detecting man-made features in aerial images," *Electronics Letters*, vol. 30, no. 7, pp. 554–555, 1994.

[6] D. L. Chenoweth, B. E. Cooper, and J. E. Selvage, "Aerial image analysis using fractal-based models," in *Proceedings of the IEEE Aerospace Conference*, pp. 277–285, 1995.

[7] B. E. Cooper, E. D. Jansing, D. L. Chenoweth, and J. E. Selvage, "Fractal error for aerial image analysis." submitted to *Pattern Recognition*.

[8] E. D. Jansing, D. L. Chenoweth, and J. Knecht, "Feature detection in synthetic aperture radar images using fractal error," in *Proceedings of the IEEE Aerospace Conference*, vol. 1, pp. 187–195, 1997.

[9] E. D. Jansing, B. S. Allen, and D. L. Chenoweth, "Edge enhancement using the fractal error metric," in *Proceedings of the 1st International Conference on Engineering Design and Automation*, 1997.

[10] E. D. Jansing and D. L. Chenoweth, "A genetic approximation of a fractal-based measure." submitted to *Evolutionary Optimization*.

[11] B. S. Allen, E. D. Jansing, J. E. Selvage, and D. L. Chenoweth, "Neural and genetic approximations of fractal error," in *Proceedings of IEEE Aerospace Conference*, 1998.

[12] C. Lin, M. Sano, and M. Sekine, "Detection of radar targets by means of fractal error," *IEICE Transactions on Communications*, vol. E80-B, pp. 1741–1748, November 1997.

[13] M. L. Brecque, "Fractal applications," *Mosaic*, vol. 17, pp. 34–48, Winter 1986/1987.

[14] K. Falconer, *Fractal Geometry–Mathematical Foundations and Applications.* New York: John Wiley and Sons, Ltd., 1990.

[15] B. B. Mandelbrot, *The Fractal Geometry of Nature.* San Francisco, CA: Freeman, 1983.

[16] H.-O. Peitgen, H. Jurgens, and D. Saupe, *Chaos and Fractals: New Frontiers of Science.* New York: Springer-Verlag, 1992.

[17] G. A. Edgar, ed., *Classics On Fractals.* Reading, MA: Addison-Wesley, 1993.

[18] B. H. Kaye, *A Random Walk Through Fractal Dimensions.* New York: VCH, 1989.

[19] P. Brodatz, *A Photographic Album For Artists and Designers.* New York: Dover, 1966.

[20] B. B. Mandelbrot, *Fractals: Form, Chance and Dimension.* San Francisco: W. H. Freeman and Co., 1977.

[21] L. J. Bain and M. Engelhardt, *Introduction to Probability and Mathematical Statistics.* Boston: PWS-Kent, 1992.

[22] J. E. Selvage, D. L. Chenoweth, and V. E. Gold, "Geometric feature extraction using the chord transform," in *IEEE Aerospace Conference*, pp. 399–405, February 1996.

Chapter 5

Appendix

Appendix A – Dissertations Supported By ONR Grant N00014-92-J-4096

FRACTIONAL BROWNIAN MOTION FOR
REPRESENTING NATURAL IMAGE TEXTURE

By

Brian Edward Cooper
B.S., University of Louisville, 1986
M.Eng., University of Louisville, 1989

A Dissertation
Submitted to the Faculty of the
Graduate School of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

Computer Science and Engineering Program
University of Louisville
Louisville, Kentucky

August 1994

FRACTIONAL BROWNIAN MOTION FOR
REPRESENTING NATURAL IMAGE TEXTURE

By

Brian Edward Cooper
B.S., University of Louisville
M.Eng., University of Louisville

A Dissertation Approved on

July 27, 1994

by the Following Reading Committee:

[signature]
Dissertation Director

[signature]

[signature]

[signature]

[signature]

ABSTRACT

Fractional Brownian motion (fBm) holds intuitive appeal for modeling natural visual textures. In addition to synthesizing artificial textures based upon fBm, fractal characteristics may be extracted from actual images, to quantify aspects of texture within them. Recent research has demonstrated that fBm may be used to model and subsequently characterize visual textures. However, the vast majority of existing investigations have relied upon the fractal dimension, variants of the fractal dimension and lacunarity, all of which assume that the observed image fits the fractal model. The error between the actual image and the fractal model should be considered. Here, the fractal error is introduced as a texture feature.

Although fractal models apply to a wide class of textures, particular emphasis is made here on aerial imagery, which few studies have included in their analyses. A localized version of the fractal error measurement was developed for the segmentation of aerial imagery. This operator was incorporated into a general purpose aerial image segmentation software package and evaluated. These successful results were compared against the previous choice of texture features for representing natural textures in aerial images, the "busyness" operator. In addition to segmentation, another important aspect of the scene evaluation process is the identification of stable features within an aerial image. Here, the fractal error also performed favorably, locating the stable, man–made objects in certain types of aerial images.

ACKNOWLEDGEMENTS

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

To the memory of my father

Joseph Leslie Cooper

CHAPTER I

INTRODUCTION AND PURPOSE

1.1 Motivation for a Texture Feature

In some black and white aerial photographs, it is possible to distinguish among the regions within the image using only the differences in their gray shade intensities. For example, trees may appear darker than surrounding fields. However, gray shade is inadequate for other images, such as the shore scene in Figure 1. Suppose that there are three categories of regions to be identified in the shore scene: water, foliage (trees) and non–foliated land. Note that the gray shades of the water and foliage overlap considerably. The image in Figure 2a reveals an attempt to isolate the foliage, but much of the water is included with the foliage in the segmented image. If all of the water is
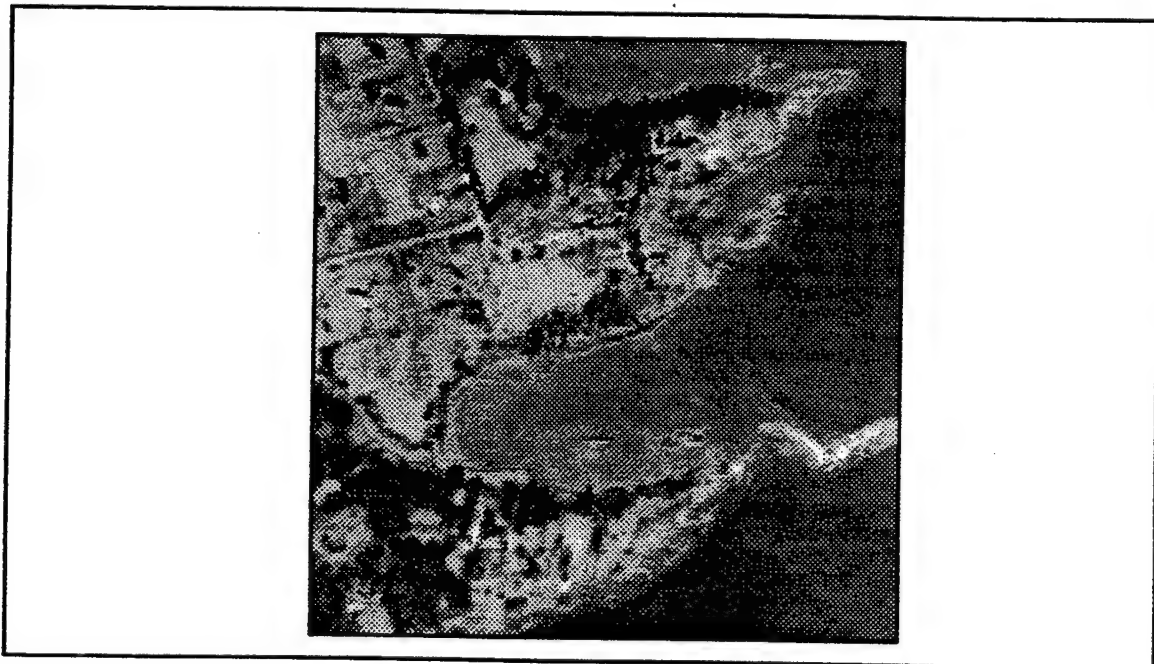
Figure 1 Sample Image: Shore Scene

(a) Highlighting the Foliage   (b) Highlighting the Water
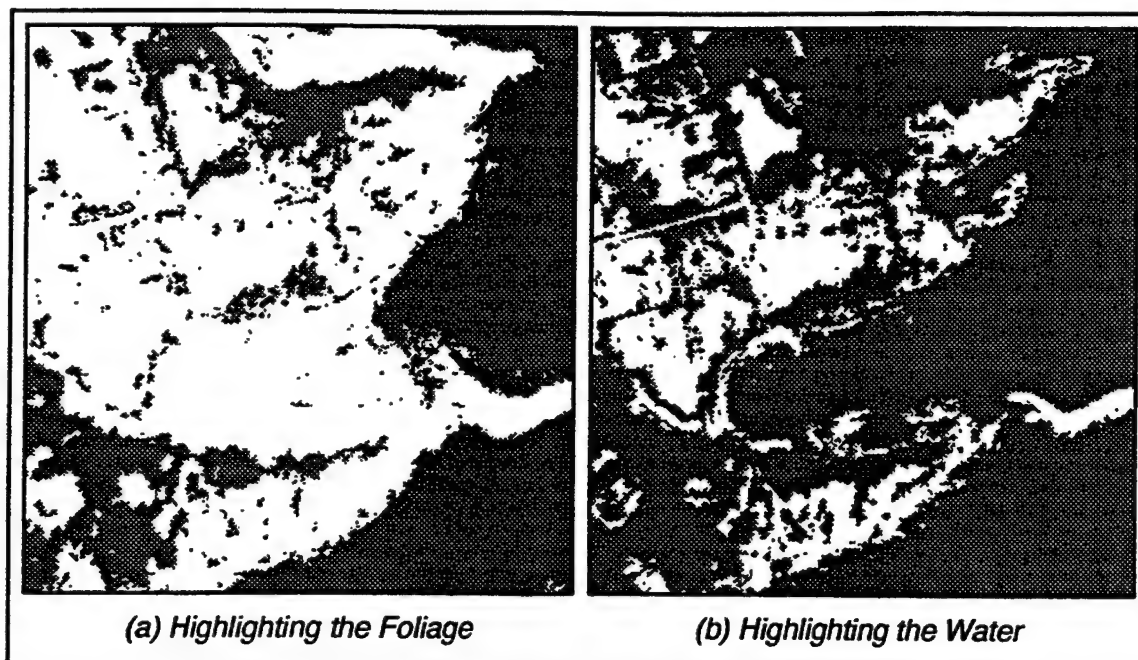
Figure 2 Segmentation of the Shore Scene Using Only Gray Shade

included, as shown in Figure 2b, then all of the foliage and portions of the non–foliated land are included with the water as well.

Although the gray shades of the foliage and water are very similar, the water tends to be smooth and uniformly shaded, whereas the foliage tends to have more variation in its intensity, or a rougher *texture*. Thus, texture may be used in combination with gray shade to characterize regions within an image. Using busyness* as the texture feature, the three regions of foliage (black), water (gray) and non–foliated land (white) may be separated, as shown in Figure 3.

In general usage, texture is usually associated with the visual and tactile perception of a material. However, in the context of image processing, only the visual aspect of texture is considered. Texture consists of two factors: the texture elements and their arrangement. The texture elements share a common size, shape or shade. These elements are arranged spatially in a particular orientation, pattern and density. Clearly, this definition includes a wide variety of textures. The natural textures which appear in

---

* The busyness texture feature is discussed later, in section 2.3.4.2.

(a) Original Image

(b) Segmented into Foliage
(Black), Water (Gray) and
Non–foliated Land (White)
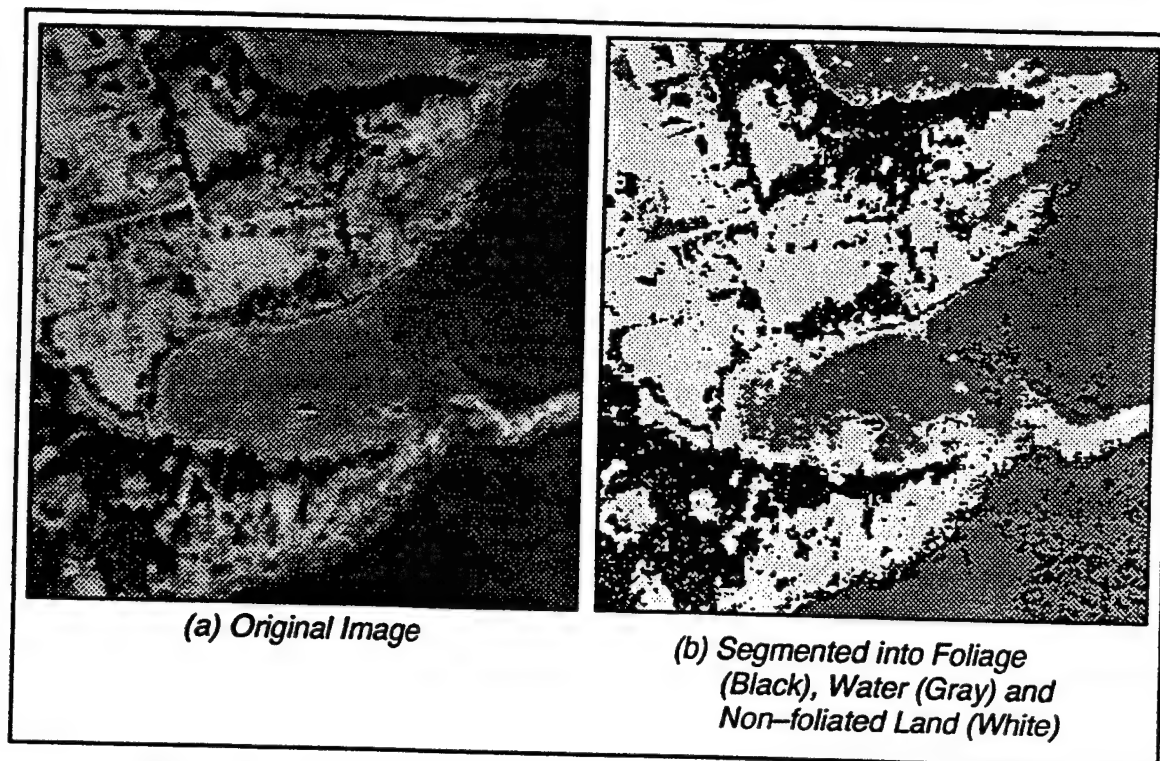
Figure 3 Segmenting the Shore Scene Using Gray Shade and Texture

aerial photographs are frequently composed of very small texture elements, often
considered to be the individual pixels which compose the image. Their arrangement
tends to be statistical and non–periodic.

1.2 Modeling Texture with Fractals

Of the many techniques designed to extract statistical, non–periodic textures with
pixel–sized texture elements, fractal methods are a fairly recent addition. Fractals may
occur in several forms, with applications ranging from data compression to fluid
mechanics to particle aggregation and erosion. The analysis of natural visual textures
was added to this list through the work of Pentland [39], who was inspired by the
prevalence of fractals in nature. Formulating fractals in terms of fractional Brownian
motion, he found a strong correlation between the fractal dimension of a surface and the
intuitive notion of roughness. The correspondence between fractals and the natural
processes which form the contents in a natural scene and the ability to quantify intuitive

aspects of texture make a fractal approach attractive for describing the variety of textures in aerial images.

1.3 Rationale for the Study

Fractional Brownian motion (fBm) holds intuitive appeal for modeling visual texture. In addition to synthesizing artificial textures based upon fBm, fractal characteristics may be extracted from actual images, to quantify aspects of the textures within them. Recent research has demonstrated that fBm may be used to model and subsequently characterize visual textures. However, most of the existing investigations are fairly narrow in scope, focusing upon a particular fractal characteristic. All of these studies assume that the observed texture fits the fractal model. The error between the actual image and the fractal model should be considered. The fractal error is introduced here as a texture feature.

Secondly, few fractal researchers have included aerial images in their studies. Accordingly, the practical analysis here places particular emphasis upon aerial images. A general purpose aerial image segmentation software package was implemented, into which the fractal error was incorporated and evaluated. Results were favorable enough to be compared against the previously chosen natural texture feature seen in section 1.1, the busyness operator. In addition to segmentation, another important goal of image interpretation is the identification of stable features, that is, objects which are less apt to change over time. The fractal error successfully located the stable, man–made objects in images of appropriate resolution.

1.4 Statement of the Problem

The primary purpose of this research was to determine fractal characteristics suitable for representing visual textures, and to develop algorithms to extract such fractal

characteristics, with particular emphasis on aerial imagery. Some questions which this dissertation addresses are given below:

- How appropriate is the fractional Brownian motion model for visual texture?

- What are some useful fBm synthesis algorithms and how do they compare?

- What are some useful fractal characteristics, how may they be estimated, and how do their extraction algorithms compare?

- How can estimation algorithms be applied?

- How may the fractal characteristics be used within a knowledge–based segmentation system?

1.5 Preview

Before discussing fractal texture representation, background material is offered on both visual texture (Chapter 2) and fractals (Chapter 3). Chapter 4 links these two topics together and examines three algorithms to synthesize fractal textures. The fractal dimension and fractal error are estimated in Chapter 5, both for describing a single texture and for segmenting an image containing multiple textures. Synthetic fractal textures, non–fractal artificial images, Brodatz textures and aerial images are included in the comparison. The success of the fractal error led to a more extensive examination of this operator for aerial image interpretation. Chapter 6 discusses two such applications of fractal error: texture segmentation and stable feature identification. The findings from the complete study are summarized in Chapter 7, along with other observations and conclusions.

CHAPTER II

A BACKGROUND ON VISUAL TEXTURE

2.1 Definition

A variety of definitions exist for the notion of texture. In general usage, texture is usually associated with the visual and tactile perception of fabric. This is reflected in the dictionary definition of texture as something composed of closely interwoven elements. In the context of image processing, only the visual aspect of texture is considered. Early attempts to define visual texture were often detailed and complicated (e.g., see Pickett [40] and Hawkins [20]), but provided the background for more recent descriptions. A particularly concise and accurate definition is offered by Hall [16]: a visual texture is "a repetitive arrangement of a basic pattern." Haralick's [18] description has the identical essence, but he elaborates upon the two aspects of texture, its primitives (i.e., the pattern) and their arrangement. A texture primitive or element (sometimes called a *texel*) is "a visual primitive with certain invariant properties which occurs repeatedly in different positions, deformations, and orientations within a given area."[1] The texels should be fairly homogeneous, sharing such properties as size, shape and coloring. The spatial dependence or interaction among these primitives could be random or dependent upon other primitives in a structural, statistical or functional manner. The resolution of a texture is related to the size of the texture elements. As the resolution increases, the texels increase in size until the observed texture completely changes character. Likewise, reducing the resolution can decrease the texel size down to a single pixel, at which point the original texture is no longer discernable.

2.2 Examples

With the above explanation in mind, it is enlightening to see how the definition of texture applies to some texture examples. A visual texture may be described by its texture primitives and their arrangement. In Figure 4a, the bricks' texture primitives are the individual bricks, arranged in a regularly repeating directional manner. The texels in

(a) Bricks	(b) Reptile Skin	(c) Grating
(d) Coins	(e) Paper	(f) Foliated Land
(g) Ice	(h) Clouds	(i) Surface of Mars
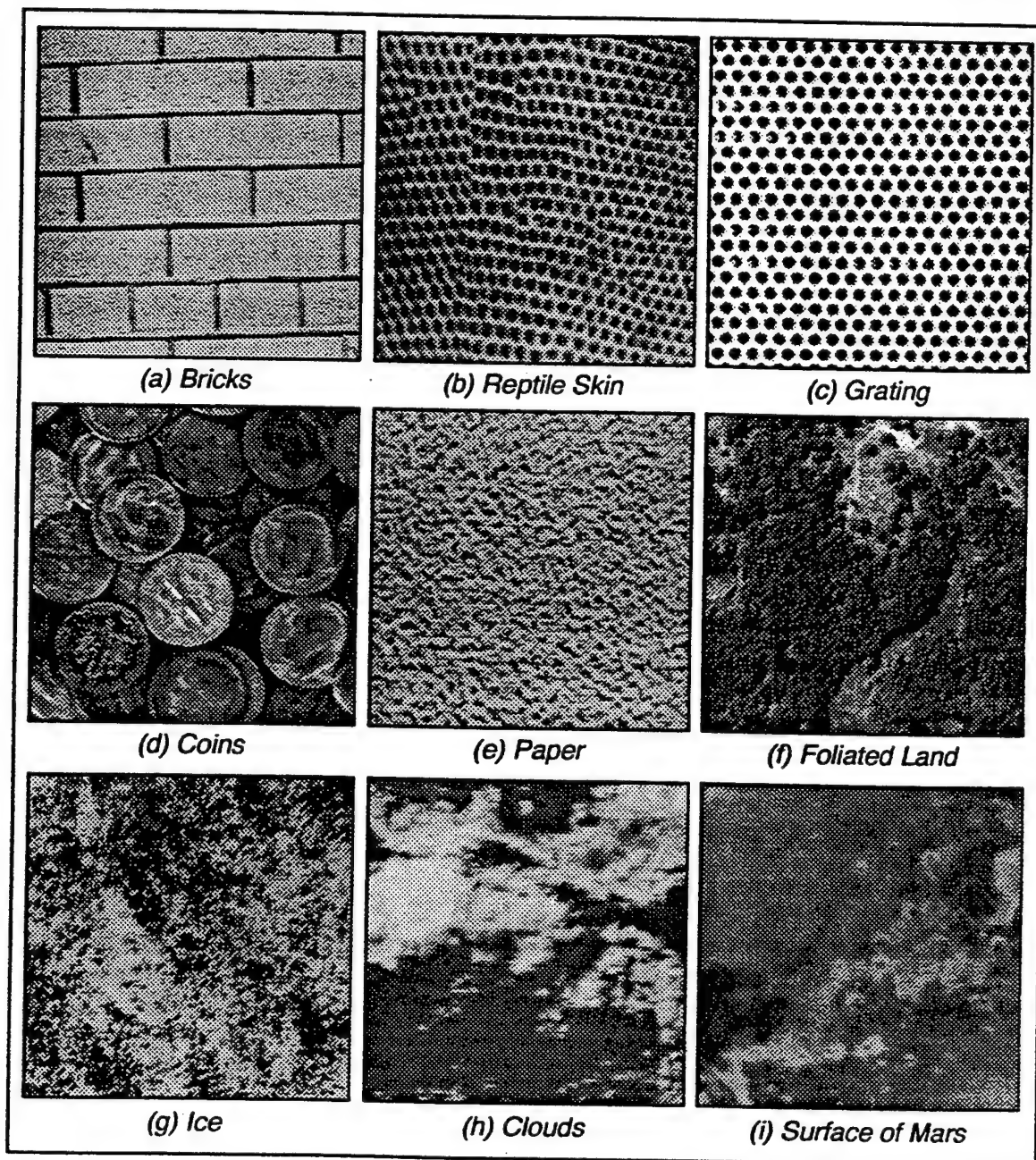
Figure 4 Texture Examples

the reptile skin texture could be considered the holes in the mesh, which are oriented in a somewhat regular fashion. The holes in the grating texture have a distinctly periodic arrangement. Randomly positioned, the coins are the obvious texels in the coins texture, although the process of extracting them may not be simple. (People have the advantage over computers of applying relevant domain knowledge, such as the physical characteristics of coins and how they may be stacked.) All of these textures have large or medium-sized texels. In contrast, there are many textures whose texels are so small that they may be considered to be the individual pixels themselves. A statistical model is often appropriate for describing the interaction of the texels. Five such examples are shown in Figures 4e through 4i: paper (magnified), foliated land, ice, clouds and the surface of Mars. Most natural phenomena in aerial photographs (e.g., fields, trees, water, ice) will have textures with very small texels.

2.3 Methods of Representation

2.3.1 Preview

A wide variety of approaches exist for representing texture. Spectral techniques consider periodicity and directionality of the texels, which are generally a few to several pixels in size. Statistical and discrete transform methods, which comprise the majority of texture approaches, are designed for pixel-sized texels with a random arrangement, as is the gray shade co-occurrence matrices method. Structural approaches concentrate upon large texels whose form follows a clearly specified structural pattern. Like the discrete transform methods, the fractional Brownian motion model is oriented toward pixel-sized texels arranged randomly. To gain perspective for the fractional Brownian motion model, each of these categories will be explained further.

2.3.2 Spectral Methods

Perhaps the earliest aspect of texture to be considered is its periodicity. Using either spatial autocorrelation or analysis in the Fourier frequency domain, the nature and degree of periodic repetition may be measured. This information can also be used to estimate the coarseness of a texture, related to the size of its texture elements.

An intuitive understanding of autocorrelation may be gained by considering two duplicate transparency copies of an image [18]. For example, picture the grating texture shown in Figure 4c. Let one of the transparencies be laid over the other, with a light source passing through the pair. As one transparency is moved across the other, the amount of transmitted light passing through will change. If the texture on the transparencies is highly periodic (as with the grating texture), then the transmitted light will vary periodically as one transparency is slid across the other.

In a discrete implementation of autocorrelation, only subregions of the image are correlated, to reduce computational effort. Otherwise, the concept is analogous to the previous description. The formula for discrete autocorrelation is given below. Let the

$$A(u, v; m, n) = \frac{\displaystyle\sum_{i=m-w}^{m+w} \sum_{j=n-w}^{n+w} X(i,j)\, X(i-u, j-v)}{\displaystyle\sum_{i=m-w}^{m+w} \sum_{j=n-w}^{n+w} X(i,j)^2}$$

gray shade at row i and column j be represented by $X(i,j)$. Typically, a square subregion is used; in the formula, the subregion is $(2W+1) \times (2W+1)$ pixels in size. The center of the first subregion is located at (m,n), while the center of the second region is offset from this by u rows and v columns. The autocorrelation $A(u,v;m,n)$ will be computed across the entire image, that is, for all (m,n). Thus, in summary, the image location is (m,n) and the amount of shift is (u,v).

Figure 5 illustrates autocorrelation applied to a simple texture at a location (m,n) and offset $(u=3, v=5)$ with $w=2$. Since the texture is periodic at that spacing, the
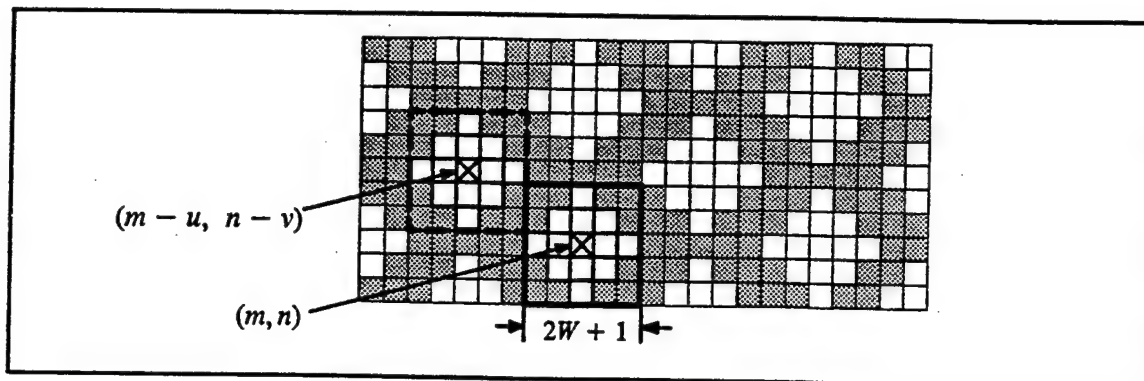
Figure 5 Example of Spatial Autocorrelation

autocorrelation function will produce the same value as it would with no offset, for every autocorrelation across the entire image. Clearly, spacings that are multiples of the period will produce the highest autocorrelation. Thus, for a known periodic texture, the spacing may be increased until this maximum is obtained, thereby determining the period. The autocorrelation may also be used to determine the coarseness of a texture. The autocorrelation will tend to decrease more slowly (at a given spacing) for large texels (coarse texture), as opposed to small texels (fine texture).

Another method of extracting information on either the periodicity or coarseness of a texture is to transform the spatial image into the frequency domain, typically using a Fourier transform. Let the power spectrum of the Fourier coefficient $F(u,v)$ be represented by $|F(u,v)|^2$. Ballard and Brown [1] identify two types of features, radial and angular, which may be measured from the power spectrum. The formulas for these two features are given in Figure 6, along with an illustration of the feature being measured on a power spectrum. The radial and angular features consider some region of interest, such as those indicated by the shaded areas. (Note that the graphs in Figure 6 are actually three–dimensional. Shown is an overhead view of the frequency plane (u,v). The power spectrum $|F(u,v)|^2$ would rise vertically from the page's surface.)

Radial measures can distinguish smooth textures (having mostly low frequencies) from rough textures (having mostly high frequencies). Likewise, periodic textures will have a peak within a particular frequency band. Angular measures can detect

$$m_{r_1,r_2} = \sum_u \sum_v |F(u,v)|^2 \qquad r_1^2 \le u^2 + v^2 < r_2^2$$

(a) Radial Features

$$m_{\theta_1,\theta_2} = \sum_u \sum_v |F(u,v)|^2 \qquad \theta_1 \le \arctan\left(\tfrac{v}{u}\right) < \theta_2$$

(b) Angular Features
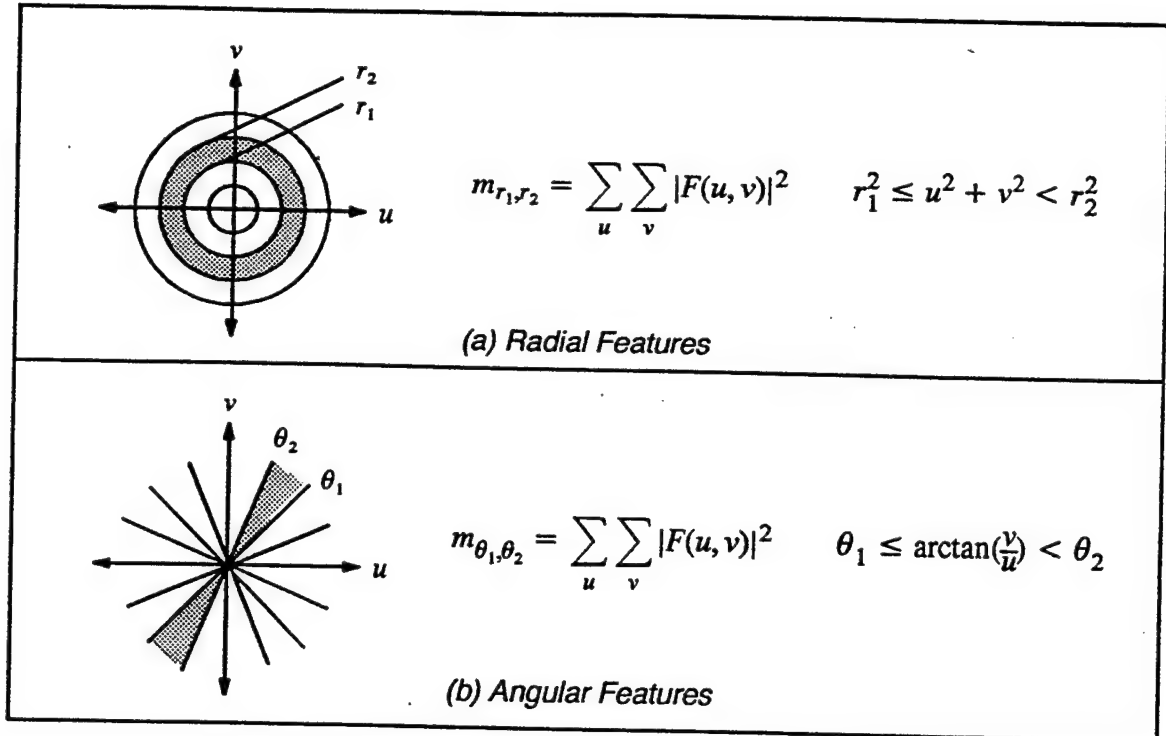
Figure 6 Texture Measurement in the Frequency Domain

directionality. If a texture has several lines or edges in the direction ϕ, then high values will occur within the frequency plane around $\theta = \phi + \pi/2$.

Spectral measures are perhaps the earliest techniques applied to image texture analysis. Predating discrete methods, Kaizer [22] judged the coarseness of aerial photographs of arctic regions using the autocorrelation function. He found a strong correspondence between the autocorrelation measurements and the coarseness rankings produced by a group of human subjects. Lendaris and Stanley [29] also took an optical approach, creating a Fraunhofer diffraction pattern of aerial photographs, equivalent to taking a two–dimensional Fourier transform. They used both the radial and angular features (with wedges spaced 9° apart) shown in Figure 6, along with measurements along parallel slits of the frequency plane, to distinguish between one–dimensional and two–dimensional regularities. Applied to a variety of aerial subimages, their technique gave inconclusive but favorable results. Kirvida [26] analyzed multispectral satellite imagery, identifying the land categories of conifers, hardwoods, open, water and urban.

He extracted texture information from the magnitude of the frequency distribution using four discrete transform algorithms. He found little difference among the transforms, but obtained an increase in classification accuracy from 73% (using only the original four spectral bands) to 98.5% when the frequency domain texture information was included. However, Hawkins [20] found considerable overlap in the spectra of aerial images of woods, urban and rural areas. He concluded that spectral measures are useful only for "rather gross discriminations," [20, p. 352] since distinct textures may have similar spatial frequency components, and sinusoidal variations in gray shade are poorly matched to textures typically found in aerial photographs. Although other applications (such as the detection and classification of coal miners' black lung disease [41]) still find spectral analysis useful, spectral methods have not been applied extensively to the analysis of aerial photography since the mid–1970's.

2.3.3 Collecting Statistics within Subwindows of an Image

Both statistical and discrete transform methods divide the image into smaller grids. These grids may or may not overlap one another, as shown in Figures 7a and 7b, respectively. Overlapping grids give better resolution, but require more computation.
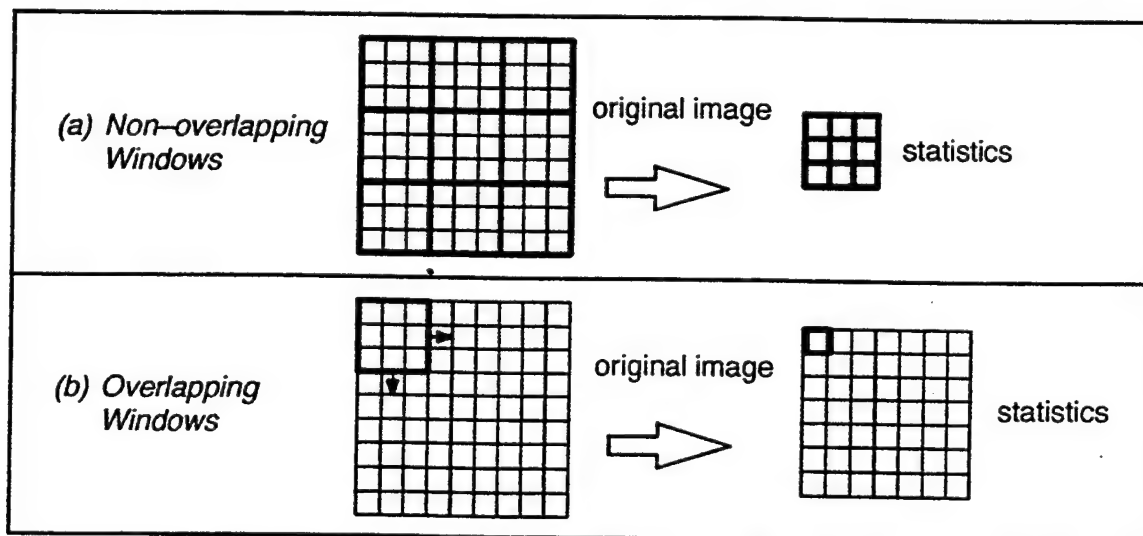
Figure 7 Collecting Statistics within Subwindows of an Image

The size of the grids, or *windows*, is subjective. Ideally, the window should be large enough to capture a texture adequately, but small enough to enclose a single texture. Within each window, various statistics may be collected on the gray shades of the enclosed pixels, such as the mean, standard deviation, skewness, minimum, maximum and range of gray shades.

For a particular statistic, the values for each window of the first row are typically stored into the first row of pixels in the output image. The values calculated from successive rows of windows form the remainder of the output image. Clearly, overlapping windows produce a larger output image than non–overlapping windows, as shown in Figure 7. (When using non–overlapping windows of size $w \times w$ on an image of size $n \times n$, the output image will be $\lfloor n/w \rfloor \times \lfloor n/w \rfloor$, where $\lfloor x \rfloor$ denotes the floor function. With overlapping windows, the output image will be $(n - w + 1) \times (n - w + 1)$, which has $w - 1$ fewer rows and columns than the original image. It is sometimes desirable to have the output image the same size as the original image. A common technique extends the original image by $w - 1$ rows and columns by copying the outside border of pixels into an extra "frame" surrounding the enlarged image.)

Although the statistics of the small regions of an image may be sufficient for certain simple textures, such an approach is often inadequate. However, when used in conjunction with discrete transforms, more favorable results may be gained, as described in the following section.

2.3.4 Discrete Transforms

Discrete transforms are used extensively in image processing and thus are natural candidates for extracting texture. Just as a value was computed from the gray shades of the pixels within each window to obtain a statistical measure, discrete transforms also use a window. However, whereas the pixels' locations within a statistical window are irrelevant, each position within a discrete transform window is specifically defined in

some manner. For example, consider a 3 × 3 window designed to highlight the edge content of an image. For convenience, let the gray shades of each pixel be represented by the letters A through I, as shown in Figure 8. The formula to the right (the Laplacian edge operator) determines the value that will be assigned to that window. As before with the statistical windows, values are computed for windows across the entire image to give an output, or filtered, image.

2.3.4.1 Edge Density

Probably the earliest representation for aperiodic textures stemmed from studying the results of edge (high–pass) filters. Regions with a proliferation of edges correspond to rough or highly textured objects. Rosenfeld and Troy [44] applied an edge operator and then binarized the image. The number of edge pixels determined the degree of roughness of the texture. Ohta [35] employed a very similar approach much later, in the context of scene interpretation. After using the same 3 × 3 Laplacian edge operator shown in Figure 8, he binarized the image at the threshold $T = mode +$ *standard deviation* × 1.4. Next, the resulting binary image was divided into 9 × 9 squares, each of which was further subdivided into nine 3 × 3 windows of pixels. If all, or all but one, of the nine 3 × 3 windows contain any edge pixels, then the center pixel (of the entire 9 × 9 square) is considered to have texture. Subsequently, this result was scanned for connected components to identify textured regions. Fortunately, Ohta provides a reasonable amount of detail on his method, as well as an example. However,

A	B	C
D	E	F
G	H	I

$$B + D - 4E + F + H$$
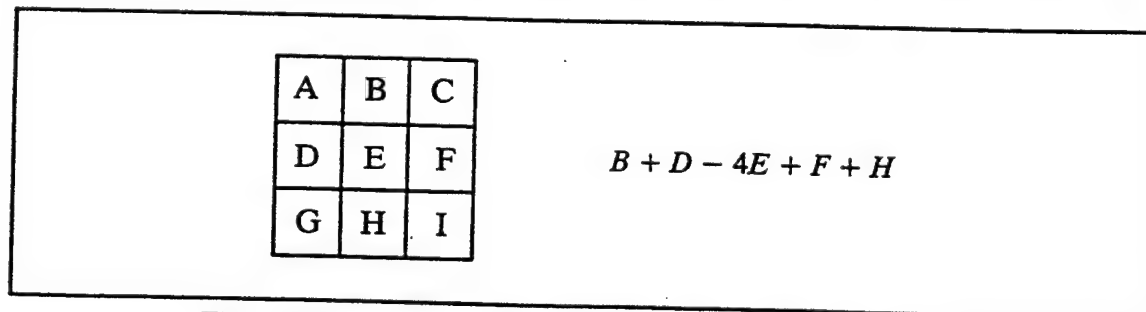
Figure 8 Specification for a 3×3 Laplacian Edge Operator

his particular scheme did not appear to perform especially well, as it significantly underemphasized the apparent textural content of the scene. In another image understanding system, Hanson and Riseman [17] used a small 2×2 edge mask (so that closely spaced parallel lines will be detected) to extract the magnitude and direction of the edge gradients. As the result tends to be highly fragmented, edges are grouped together according to the similarity of their magnitudes and gradients. If all of the longer line segments are removed (by thresholding), the textured regions may be found. (Likewise, keeping only the longer line segments of high magnitude yields the primary structural components of the image.)

2.3.4.2 Busyness

Originally developed by Rosenfeld and Kak [43], the busyness operator measures the roughness of a texture by the minimum of the gray shade differences in the horizontal and vertical directions. Busyness uses a single filter, whereas the edge density method requires binarization and counting the number of edge pixels in a region after applying the edge filter. In addition to being faster, busyness eliminates the subjectivity associated with determining the binarization threshold and the density measurement. The gray shades of the pixels within a 3×3 window are represented by the letters A through I, as shown in the right of Figure 9. The average absolute gray shade difference is calculated horizontally (V_X) and vertically (V_Y), as shown in Figure 9a. The minimum of these two values is the busyness of the pixel at the center of the window.

The average gray shade difference in the direction perpendicular to an edge will be high. In order to reduce this undesirable emphasis, the minimum of the horizontal and vertical averages is taken. (Although it will always be present to some degree, the edge density method can remedy this problem by considering the density of edges within a sufficiently large area.) However, regions containing diagonal edges will be highlighted. A pair of diagonal averages were added to the busyness formulation in [5] to alleviate

(a) Original Definition			
$V_X = (|A - B| + |B - C| + |D - E| + |E - F| + |G - H| + |H - I|)/6$ $V_Y = (|A - D| + |D - G| + |B - E| + |E - H| + |C - F| + |F - I|)/6$ Busyness $= \min(V_X, V_Y)$	A	B	C
(b) Modified Definition $V_X = (|A - B| + |B - C| + |D - E| + |E - F| + |G - H| + |H - I|)/6$ $V_Y = (|A - D| + |D - G| + |B - E| + |E - H| + |C - F| + |F - I|)/6$ $V_{D1} = (|D - B| + |G - E| + |E - C| + |H - F|)/4$ $V_{D2} = (|B - F| + |A - E| + |E - I| + |D - H|)/4$ Busyness $= \min(V_X, V_Y, V_{D1}, V_{D2})$	D	E	F
	G	H	I

Figure 9 Busyness Operator

this problem. The modified formula is given in Figure 9b. Although edges at angles other than the horizontal, vertical and diagonal directions can still cause artificially high busyness values, including the diagonal measures significantly improves the quality of the busyness operator.

2.3.4.3 Texture Energy

The concept of texture energy originated from Laws' study [28] of statistical and discrete transform methods of extracting texture information. He analyzed a wide variety of texture features, including gray shade co-occurrence matrices, correlation methods and statistical–spatial methods. In the latter category, texture was extracted in two steps: small masks (3 × 3 or 5 × 5) were convolved with the image, and then local statistics were collected from within larger windows (15 × 15 or 31 × 31). The filtering mask should be small enough to capture only a single texture, and the statistical window should be large enough to get an adequate sampling of the texture. From this study, Laws found that a set of ad hoc masks designed for detecting spots, lines and other small features worked best for the filtering masks. The most effective statistical measure was the standard deviation (or equivalently, the variance).

Since the variance is related to the "energy" of a signal, and the filters attempt to extract a texture signal, Laws called such an approach a "texture energy measure." Combining both discrete filtering and the calculation of local statistics, he developed and refined his initial study into his final texture energy technique. A pictorial interpretation of the texture energy model is offered in Figure 10. Four one–dimensional ad hoc filtering masks (see Figure 11) compose the final model: Level, Edge, Spot and Ripple.
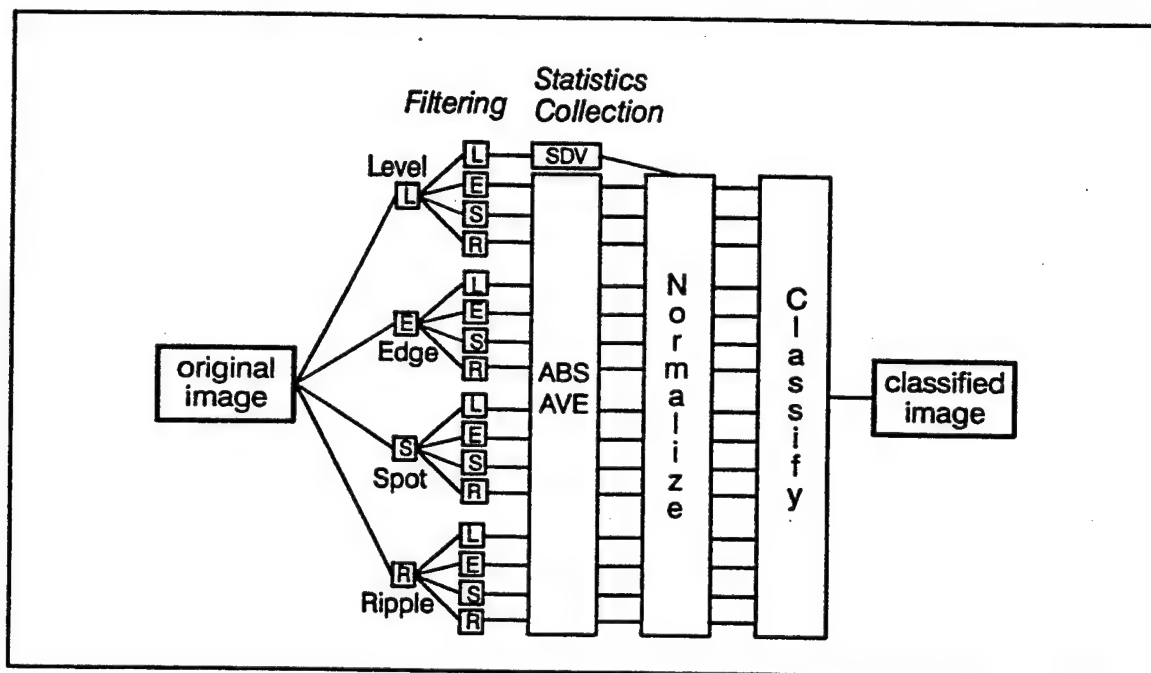
Figure 10 Texture Energy Model

LEVEL: [1   4   6   4   1]
EDGE: [−1 −2   0   2   1]
SPOT: [−1   0   2   0 −1]
RIPPLE: [1 −4   6 −4   1]

(a) One–dimensional Masks

Edge−Spot

$$\begin{bmatrix} 1 & 0 & -2 & 0 & 1 \\ 2 & 0 & -4 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ -2 & 0 & 4 & 0 & -2 \\ -1 & 0 & 2 & 0 & -1 \end{bmatrix}$$

(b) Example of a Two–dimensional Mask formed by Convolving the One–dimensional Masks Edge and Spot

Figure 11 Convolution Masks for Texture Energy

Convolving each of these four masks both horizontally and vertically yields sixteen two–dimensional masks, one of which is shown in the right of Figure 11. (Equivalently, if each of the one–dimensional masks is considered to be a $n \times 1$ vector, then each $n \times n$ mask is created by multiplying a one–dimensional mask with the transpose of another one–dimensional mask.) Thus, sixteen filtered images are produced. One of these, the Level–Level image, is used to eliminate contrast variation among the remaining fifteen images. In the second stage, a statistical value is obtained from within a 15×15 window surrounding each pixel of the filtered images. The standard deviation is measured locally across the Level–Level image, whereas the absolute average is calculated for the other fifteen images since the absolute average is a faster, but less accurate, approximation of the standard deviation. The resulting images, called "texture energy planes," are used as features to classify the pixels of the image.

High classification accuracies were achieved using texture energy on a collage of various sized blocks (16×16, 32×32 and 128×128) of eight Brodatz [3] textures. Without criticizing the quality of the texture energy features themselves, Hsiao and Sawchuk [21] note that Laws' segmented images contain "considerable errors in the interior of the large regions" and poor performance "near the borders between textures." When the statistical window encloses a single texture, its value will tend to be accurate. However, when this window overlaps a texture boundary, thus enclosing two or more textures, its value may be unreliable. Hsiao and Sawchuk examined the variance in the texture energy (i.e., the variance of the absolute average of the filtered image) to determine texture borders more effectively, as a high variance in texture energy often indicates a border. Scattered, small clusters of misclassified pixels were treated using probabilistic relaxation.

Whereas both Laws [28] and Hsiao and Sawchuk [21] concentrated upon collages of Brodatz textures, Cooper [6] applied texture energy to aerial images. Gray shade is often the single most important feature for distinguishing regions within aerial images.

On the other hand, the Brodatz textures in the other two cited studies had approximately the same mean gray shade, and must therefore be separated solely by the differences in their textural properties. Although the segmentation of aerial images sometimes benefits from textures which are separable by gray shade, the textures in aerial images also possess much greater variability in content than do the highly homogeneous collages of Brodatz textures. Thus, a combination of gray shade and busyness gave better segmentation results on aerial imagery than texture energy alone. A combination of gray shade and texture energy was not examined in [6], since the anticipated extra increase in quality of the texture energy approach over busyness did not warrant the additional computational burden.

2.3.5 Gray Shade Spatial Dependence Co-occurrence Matrices

The gray shade spatial dependence co-occurrence method* considers the statistical distribution of pairs of pixels separated by a particular fixed distance. Let the distance between a pair of pixels $I(j,k)$ and $I(m,n)$ be expressed in polar form, with a distance r and angle θ from the horizontal axis, as shown in Figure 12a. (Obviously, the selection of the spacing (r,θ) will be restricted by the discrete nature in which is the image is stored.) The image (or region of an image) is examined to determine the number of times that each pair of gray shades occurs at the given distance and angle. This results in a matrix of size $G \times G$ (shown in Figure 12a), where G is the number of gray shades in the image. This matrix (after normalization) represents the observed statistical distribution of the gray shades in the image at one particular spacing.

---

* A collection of similar terms exists for the gray shade spatial dependence co-occurrence method, allowing such variations in:
  (1) using "gray shade," "gray level" or "gray tone"
  (2) the placement or omission of "spatial"
  (3) the omission of either "dependence" or "co-occurrence" (but not both)
  (4) the inclusion of either "matrix" or "statistics," to emphasize these aspects of the technique
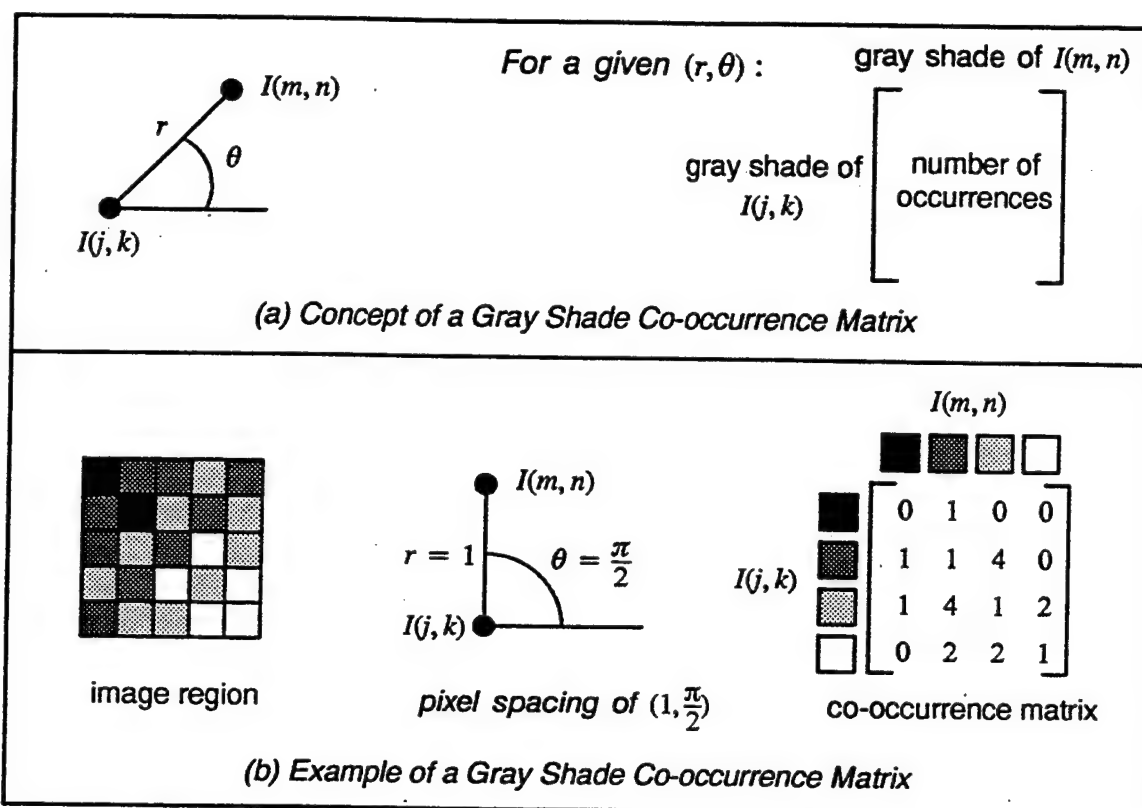
Figure 12 The Gray Shade Spatial Dependence Co-occurrence Method

Figure 12b offers an example of a 5×5 image region which has been quantized into four gray shades. The matrix at the far right contains the number of occurrences of each gray shade that is immediately above (i.e., a pixel distance of one at a ninety degree angle) every other gray shade. Such a matrix must be generated for every spacing (r, θ) of interest.

Although the contents of the co-occurrence matrix may be used directly as textural features [45], typically simple statistical features are computed from the matrix. These features vary both in form and name; a few of the more common ones [15, 18, 45] are included in Figure 13. (Assume that the matrix is $G \times G$, with array indices ranging from 0 to $G - 1$. Let $p_{x,y}$ be the probability of a pixel with gray shade x being located at a spacing (r, θ) to a pixel of gray shade y, calculated by dividing the matrix element (x,y) by the sum of the matrix elements.)

Uniformity or Energy	$\displaystyle\sum_{x=0}^{G-1}\sum_{y=0}^{G-1}(p_{x,y})^2$
Entropy	$\displaystyle-\sum_{x=0}^{G-1}\sum_{y=0}^{G-1}p_{x,y}\log(p_{x,y})$
Maximum probability	$\max(p_{x,y})$
Local homogeneity*	$\displaystyle\sum_{x=0}^{G-1}\sum_{y=0}^{G-1}\frac{p_{x,y}}{1+|x-y|^2}$
Inertia*	$\displaystyle\sum_{x=0}^{G-1}\sum_{y=0}^{G-1}|x-y|^2 p_{x,y}$

* Other features may be obtained using a different exponent

Figure 13 Examples of Features Extracted from a Co-occurrence Matrix

Note that these features do not necessarily correspond to any intuitive notions of textural properties. "Even though these features contain information about the textural characteristics of the image, it is hard to identify which specific textural characteristic is represented by each of these features." [19, p. 144] Fine textures will tend to have a co-occurrence matrix with fairly equal values, whereas a coarse texture (i.e., with larger texture elements) is more apt to have a localized distribution, so that most of the occurrences will occupy one particular portion of the co-occurrence matrix. The uniformity feature (see Figure 13) is lowest when the probability is evenly spread; the opposite is true for entropy. The maximum probability gives some insight into the coarseness of the texture. When the larger probabilities are located along the diagonal of the matrix, the local homogeneity will be high, while the inertia will be low.

The gray shade spatial dependence co-occurrence method is a popular choice for analyzing natural images. Haralick et al. [19] applied the technique to photomicrographs of sandstone, aerial photographs and multispectral satellite images. Images were

quantized to either eight or sixteen gray shades, and matrices were generated from either 20×50 or 64×64 sized regions. They narrowed the pixel spacings to four, using a distance of one pixel ($r = 1$) and four angles ($\theta = 0°, 90°, 180°, 270°$). Fourteen features were computed for each of these four spacings. Although the same four pixel spacings were always used, the feature selection was tailored to the category of images. The mean and variance of four features for the photomicrographs yielded a classification accuracy of 89% with six classes. The aerial photographs used eleven features, but each of the four pixel spacings were pooled together to give a non–directional mean, range and deviation for each of the eleven features. With eight classes, 82% accuracy was obtained. The mean variance of four features, combined with eight spectral features, allowed 83% of the multispectral images (with seven classes) to be classified correctly.

Conners et al. [4] introduced two new features, cluster shade and cluster prominence, to provide a means of separating certain types of visually distinct textures that other features could not distinguish. These two features were combined with four others, each of which was extracted initially from 48 spacings ($r = 1, 2, 4, 6, 8, 12, 16, 20$ and $\theta = 0°, 19°, 75°, 90°, 109°, 165°$). They carefully determined the region size and types of training samples to use, oriented toward high resolution aerial images. Using these features, image regions were identified as uniform, boundary or unspecified. The latter two region types were successively split to form labeled uniform regions. Their procedure achieved classification accuracies between 83% and 90%, identifying nine classes within the aerial images. Trivedi and Harlow [45] employed the same six features to locate regions within an aerial image which contain certain small objects of interest. However, they supplemented these features with the individual elements of the co-occurrence matrices themselves, making it unclear why the six features need to be computed.

Although the gray shade spatial dependence co-occurrence method has been applied successfully to aerial imagery, its computational demands must be faced. Each

pixel spacing determines a co-occurrence matrix, and each matrix yields a set of descriptive features. Clearly, there are a vast number of potential pixel spacings to measure within an image region, and even the subset of these spacings which is useful can still remain large. Several features may be extracted from each of the resultant co-occurrence matrices. Since it may be difficult to match these features to obvious visual characteristics, it may be difficult to determine which features are suitable until they have been implemented and applied to a representative sample of images. Typically, only a few spacings and features are proposed, and then sample images are analyzed to determine a useful subset of these to include in the final model.

2.3.6 Stochastic Methods

The stochastic approach is related to the co-occurrences method, but it is a more general representation. It considers texture to be a sampling from a probability distribution. More specifically, the gray shades of the pixels in a rectangular image are represented by a $N_1 \times N_2$ lattice L of random variables, called a *random field*:

$$L = \{(i,j) : 1 \le i \le N_1, \ 1 \le j \le N_2\}$$

Each pixel (i,j) in the lattice L is mapped to a gray shade between 0 and $G - 1$ by a function X, so that $X(i,j) = g$ for some gray shade g. The *neighborhood* $n_{i,j}$ of a pixel (i,j) is typically a set of pixels which are close to (i,j) (see Figure 14), although proximity is not a requirement. Precisely, a neighborhood $n_{i,j}$ cannot contain the pixel (i,j), and if
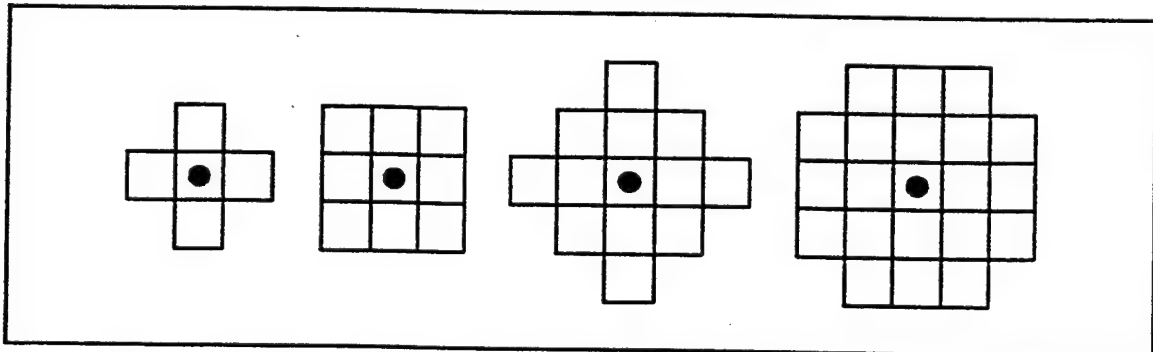
Figure 14 Typical Neighborhoods around a Pixel, of Increasing Size

one pixel is in the neighborhood of another, then the other pixel must be in the neighborhood of the first. Finally, a random field is a *Markov random field* when the probability of pixel (i,j) given all other pixels in the lattice is equal to the probability of pixel (i,j) given only those pixels in the neighborhood of (i,j):

$$P(X(i,j)|X(1,1),...,X(i,j-1),X(i,j+1),...,X(N_1,N_2)) =$$
$$P(X(i,j)|X(k,l) \text{ for } (k,l) \in n_{i,j}) \text{ for all } (i,j) \in L, \text{ and } P(X(i,j)) > 0$$

Whereas a Markov random field is characterized by conditional probabilities, joint probability distributions may be used in an equivalent representation, called a *Gibbs random field*.

Typically, the goal is to create an effective texture model, and then attempt to fit actual textures to that model. Pratt et al. [42] represented the gray shade of each pixel by a Gaussian distribution; Cross and Jain [9] used a binomial distribution. In both studies, synthetic textures were generated from the Markov random field model. Cross and Jain provide extensive examples of artificial textures created from their binomial distributed Markov random field. The binomial parameter θ (representing the probability of a success) is formulated in terms of a weighted sum of the gray shades of a pixel's neighboring pixels, where each weight corresponds to the pixel's particular location in the neighborhood. By adjusting these weights, Cross and Jain were able to control textural characteristics such as directionality, the sizes and types of clustering, and the sharpness of the gray shade transitions. Tested on twelve Brodatz [3] textures, their model worked well for representing fine–grained homogeneous textures. However, larger grained textures, especially those with large areas of equal brightness, posed difficulties, as was the case with textures having "strong regularity or cloud–like inhomogeneities." [9, p. 36] Also, a large sample is required to obtain reliable estimates for the model.

In addition to modeling the textures themselves, Derin and Cole [12] also formulated texture segmentation in terms of a Gibbs random field. In order to become

computationally tractable, their original maximum a posteriori estimation was simplified and approximated by a suboptimal version. Their segmentation algorithm was applied to artificially created images consisting of two or three textures, quantized to either two or four shades of gray. However, they did not approach the complexities related to the greater number of textures and wider range of gray shades present in aerial images.

2.3.7 Structural Methods

Whereas the other texture representations discussed so far are suited toward small to medium sized texels, structural methods are designed to work with large, clearly identified texture elements. It is usually assumed that the texels are extracted and defined beforehand. It should be noted that, although this task may often be straightforward, identifying the texels may often be very challenging, as was mentioned with the coins texture in Figure 4.

Furthermore, structural methods depend upon the texels forming a repeating pattern (i.e., a regular structure), which can be described by a set of rules. The rules are often included within a formal *grammar,* similar to the way that a compiler describes a computer language by specifying a set of symbols and the rules by which these symbols may be combined. For a computer language, the symbols consist of characters such as letters, numbers and operators. For a texture, the symbols will be the texels. The rules are specified such that the left hand symbol may be replaced by the symbol(s) on the right hand side, where the left and right sides are separated by an arrow. These rules may be applied as many times as needed. For example, a simple brick pattern could be described by the two rules in Figure 15a. The top rule creates a single row of bricks; the bottom rule specifies how additional rows may be added. Another brick pattern is given in Figure 15b, along with a corresponding set of rules. (Note that, for either texture, alternate rule specifications are possible.) A texture may be recognized by applying the rules backwards until the initial symbol is obtained.
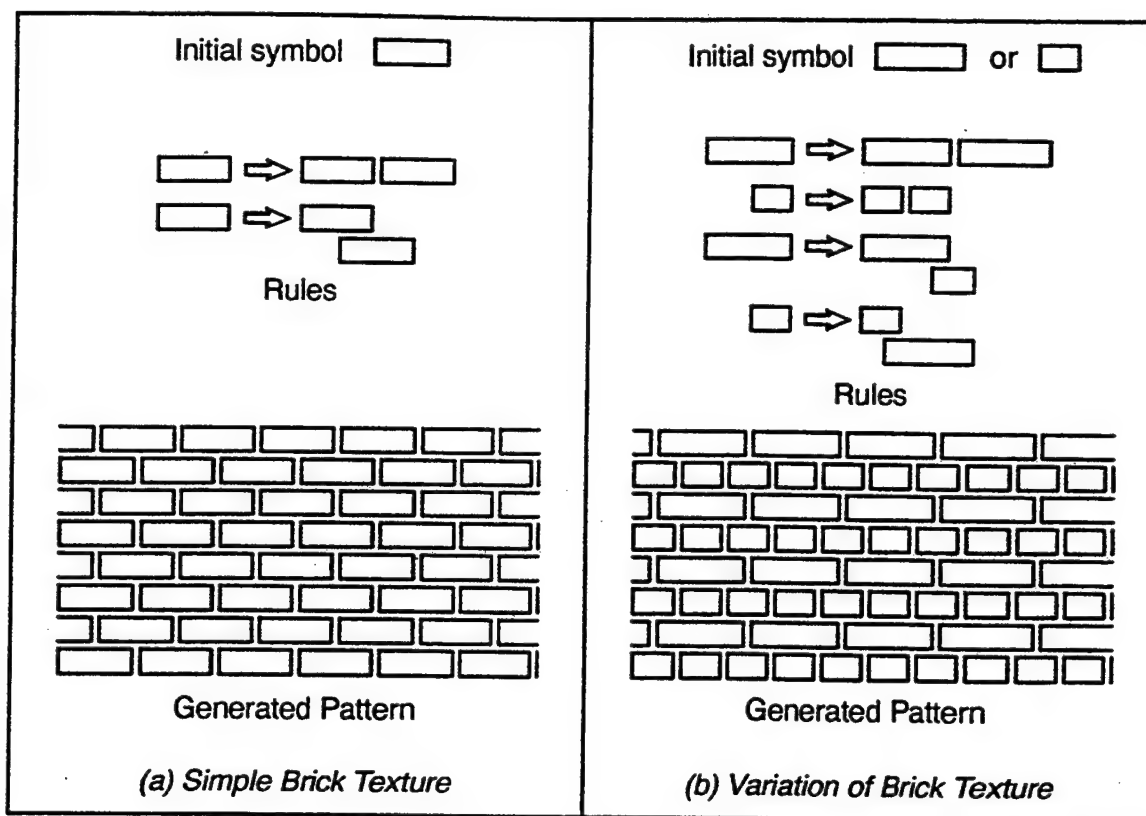
Figure 15 Example of a Shape Grammar

Probabilities may be assigned to the rules, to form a *stochastic grammar*. For example, consider a brick pattern with rows of large bricks and rows of small bricks. Suppose that there are no restrictions about how many rows of large bricks there must be between rows of small bricks, or vice versus (as opposed to the strict alternation in Figure 15b). However, let there be twice as many rows of large bricks overall as rows of small bricks. Then this property could be reflected by giving a higher probability to the formation of a new row of large bricks and a small probability to the rule which generates a new row of small bricks. The reptile skin texture (Figure 4b) may be generated using a stochastic grammar [1]. The underlying tessellation of this texture is extracted, and the dual graph of the tessellation pattern is used to form the symbols of the grammar.

Ballard and Brown [1] identify three types of grammars, each of which may presumably be made stochastic: shape, tree and array. The symbols used by a shape

grammar correspond to the texels. The grammars described previously (i.e., for the bricks and reptile skin textures) are all shape grammars. For a tree grammar, a tree structure is embedded within the image array. This structure defines the top level arrangement of the texels. After applying the top level of rules, a lower level of rules may be used to describe the pattern of pixels which define each texel. An array grammar is similar to a tree grammar. However, it uses the implicit structure of the image array rather than embedding a tree. As with the tree grammar, rules may be arranged in hierarchical levels corresponding to the different levels of resolution within the texture.

Formal grammars may be developed for textures exhibiting a high degree of regularity, such as a brick wall, wire fence or herringbone pattern. Imperfections or variations in the structural arrangement may be accommodated by making the grammar stochastic, as was done for the reptile skin texture. Although such regularity can occur in certain types of aerial imagery (such as an urban or suburban scene), often a definitive placement of texels is absent. Also, the texels in aerial imagery may be too small or too difficult to extract to be used with a formal grammar. Thus, structural methods are seldom applied to aerial photographs.

2.3.8 Fractal Methods

2.3.8.1 Fractional Brownian Motion

First popularized by Mandelbrot in [31], fractals may occur in several forms, with applications ranging from data compression to fluid mechanics to particle aggregation and erosion. In the analysis of natural visual textures, the model most often used is fractional Brownian motion, introduced by Mandelbrot and Van Ness [34]. Fractional Brownian motion (abbreviated *fBm*) is an extension of ordinary Brownian motion, which refers to the erratic motion of small suspended particles, resulting from random collisions with other particles.

Early research synthesized natural terrain surfaces using wire–mesh terrain maps of two–dimensional fractional Brownian motion. Pentland [39] extended fBm to image textures in general. By mapping the height of a two–dimensional fBm signal to a gray shade (dark is low and light is high), natural cloud–like image textures may be produced. The fractal dimension of the signal corresponds closely to the perceived roughness of the visual texture. For example, Figure 16 shows a series of two–dimensional fBm images with increasing dimension and roughness. Texture representation is only one relatively minor aspect of the burgeoning field of fractal geometry. The following chapter provides the background needed to understand how fractals may be used to represent visual texture.

(a) Dimension 2.3   (b) Dimension 2.5   (c) Dimension 2.7
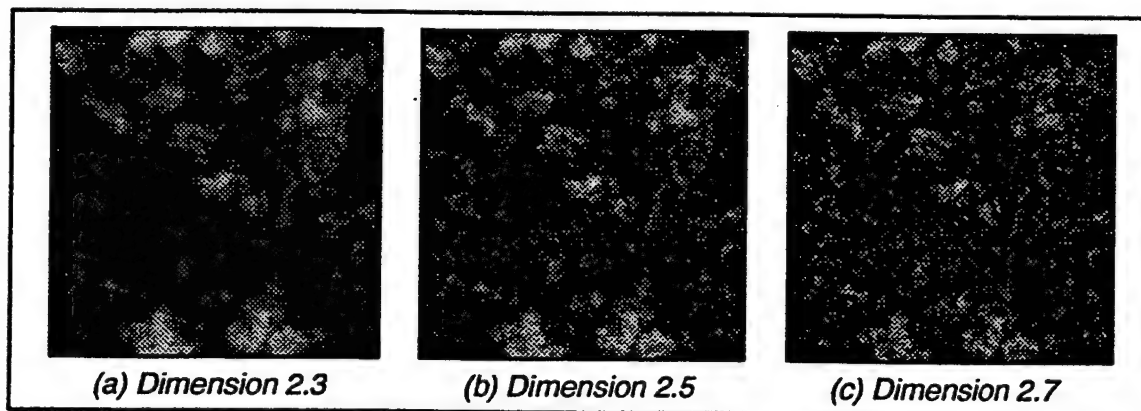
Figure 16 Examples of Two-Dimensional Fractional Brownian Motion

CHAPTER III

A BACKGROUND ON FRACTALS

3.1 Introduction

Traditionally, geometrical descriptions have been based upon Euclidean geometry and Plato's Ideal Forms. While these concepts have been useful for describing many natural phenomena (such as the elliptical model of planetary orbits), objects such as lines, planes, cubes and spheres are usually found in artificial, not natural, environments. The "ideal" shapes fail to describe a wide spectrum of natural processes, ranging from the microstructure of fractured materials to the formation of galactic clusters. However, fractal geometry readily accommodates the complexity and infinite detail present in nature, such as the crumpled surface of mountainous terrain or the flow patterns in turbulent fluids. "Physical processes that modify shape through local action... will, after innumerable repetitions, typically produce a fractal surface shape." [39, p. 662] As observed by Mandelbrot [33], fractals appear to be the "Geometry of Nature."

3.2 Definition

While recognizing fractals in the natural world is fairly simple, developing a definition of a fractal is much more difficult. The term *fractal* was first used by Mandelbrot [31], derived from the Latin *fractus,* meaning "fragmented" or "irregular." Attempting to provide more precision, he offers a tentative definition [33] which considers a fractal to be a set whose Hausdorff–Besicovitch dimension is strictly greater than its topological dimension, that is, "a set for which the only consistent description of its metric properties requires a 'dimension' value larger than our standard, intuitive definition of a set's 'dimension.' " [39, p. 662] This definition, although very specific, is

overly restrictive. Recognizing that fractals may occur in many forms, La Brecque gives a less technical description, that a fractal "has a rough shape to one degree or another made of parts which, when magnified, resemble the whole." [27, p. 34] The reader of fractal literature frequently senses a reluctance to confront the elusive task of developing a precise (and often inadequate) definition. For example, Barnsley [2] prefers instead to guide the reader to an intuitive understanding of fractals through examples, descriptions and usage. However, Falconer [13] faces the challenge directly, opting to describe the properties of a fractal, rather than use a definition, so as not to be too exclusive. (Falconer makes the analogy between defining a fractal and defining life: both are better described by their characteristics. It seems an appropriate analogy for a concept that corresponds so well to natural processes.) He refers to the set F as a fractal if it has properties such as those listed below (see [13], pp. xx–xxi):

1. F has a fine structure, i.e., detail on arbitrarily small scales.
2. F is too irregular to be described in traditional geometrical language, both locally and globally.
3. Often F has some form of self–similarity, perhaps approximate or statistical.
4. Usually, the 'fractal dimension' of F (defined in some way) is greater than its topological dimension.
5. In most cases of interest F is defined in a very simple way, perhaps recursively.

Clearly, this description incorporates the other definitions. The second property includes Mandelbrot's origin of the term *fractal;* the fourth repeats his original definition. La Brecque identifies characteristics related to the first and third properties. Rather than explaining the aspects of Falconer's description (such as "self–similarity" and "fractal dimension") in this section, these terms are better understood in the context of examples, as illustrated in the following sections.

3.3 Self–Similarity and Self–Affinity

3.3.1 Discussion

Falconer's third property of fractals mentions "some form of self–similarity." Note that in older publications, "self–similarity" sometimes refers collectively to both self–similarity and self–affinity. However, these terms have distinct meanings. Strict self–similarity requires that an object consist of smaller but otherwise exact copies of itself. Such fractals will have the identical appearance under any magnification. The self–similarity may be exact or statistical, in which the pattern of the fractal is repeated in an exact fashion (exact self–similarity) as opposed to using a random element to position or modify the repeated pattern (statistical self–similarity). Other fractals scale the pattern according to the magnification level, called *self–affine* fractals. Due to this scaling, self–affine fractals will reveal differences under various magnification levels. Like self–similar fractals, self–affine fractals may be either exact or statistical in form.

Although self–similarity may occasionally be used generically for either self–similarity or self–affinity, self–similarity is actually a special instance of self–affinity. For a self–similar fractal with E coordinates, a uniform scaling constant r may be applied:*

$$\mathbf{x} = (x_1, ..., x_i, ..., x_E)$$
$$R_S(\mathbf{x}) = (rx_1, ..., rx_i, ...rx_E)$$

However, self–affinity allows each element to have a unique scaling constant:

$$\mathbf{x} = (x_1, ..., x_i, ..., x_E)$$
$$R_A(\mathbf{x}) = (r_1x_1, ..., r_ix_i, ...r_Ex_E)$$

Thus, self–similarity results from self–affinity when the scaling parameter is the same for all coordinates. Note that these definitions may be applied to the graphs of functions of an independent variable t. For example, consider a function $y(t)$. Two coordinates

---

* Note that rotation, reflection and translation are permitted for both self–similarity and self–affinity. However, these properties are ignored in this discussion, to focus on the scaling differences.

$(E = 2)$ are present, expressed as (t,y) using the above notation. With self–similarity, a scaling of r along the t axis produces the same scaling along the y axis: $R_S(t,y) = (rt, ry)$. However, with self–affinity, the scaling may be different: $R_A(t,y) = (r_1 t, \; r_2 y)$.

Of particular interest is a special case of self–affinity in which the scaling factors of the independent variables are all equal to one another (call this value r), and the scaling factors of the dependent variables are all equal to one another (call this value r_y). Furthermore, the relationship between the two constants is given by the equation $r_y = r^a$. Using the same two–variable example, this particular self–affinity produces the pair $(rt, r^a y)$. This type of self–affinity leads to the development of *fractional Brownian motion,* a generalization of Brownian motion, which describes the random movement of a particle subject to collisions and other forces. Fractional Brownian motion will be examined further in section 3.3.4.

3.3.2 Examples of Self–Similar Fractals

Perhaps the earliest known fractal is the Cantor set, developed by Georg Cantor in 1883 while he was investigating the properties of continuous sets [10]. The most familiar of the Cantor sets is the middle third Cantor set, otherwise known as the triadic Cantor set or ternary Cantor set (see Figure 17), which is an exactly self–similar fractal. The process begins with the closed line interval [0,1]. The central third $(\frac{1}{3}, \frac{2}{3})$ of this interval is removed, leaving the two intervals $[0, \frac{1}{3}]$ and $[\frac{2}{3}, 1]$, shown as the first iteration in Figure 17. The central third of each of these intervals is similarly removed, leaving the intervals $[0, \frac{1}{9}]$, $[\frac{2}{9}, \frac{1}{3}]$, $[\frac{2}{3}, \frac{7}{9}]$, $[\frac{8}{9}, 1]$. The Cantor set is the result of an infinite number of these deletions. Although it may seem that nothing remains, the Cantor set in fact contains an uncountably infinite number of points, in spite of the fact that it has zero
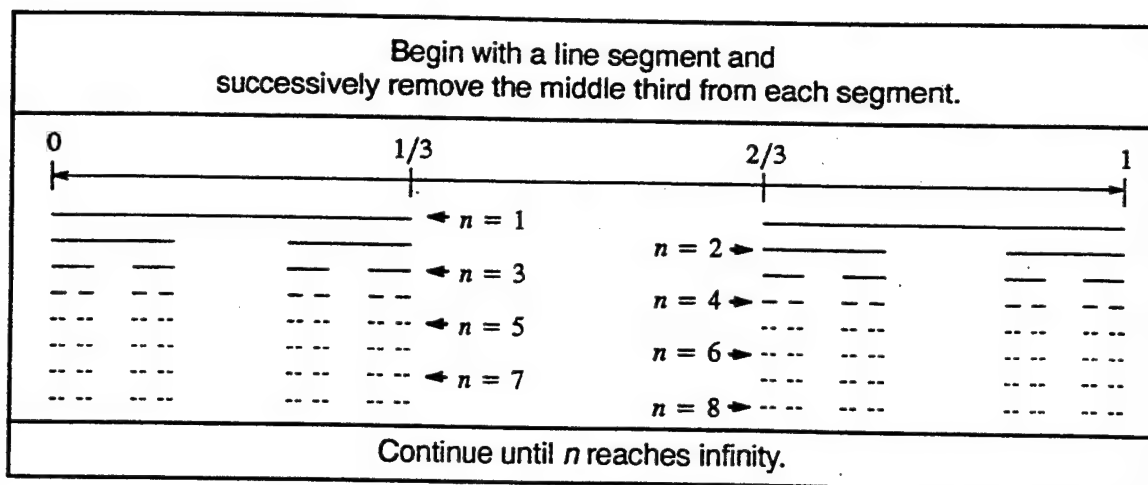
Figure 17 Construction of the Ternary Cantor Set

length [10]. (The length of the set of intervals at the nth step of the ternary Cantor set is $(2/3)^n$, whose limit approaches zero as n approaches infinity.)

Originated by von Koch in 1904, the Koch curve is another pathological structure which challenged classical mathematics [31]. Although it is continuous, it has no tangents anywhere and thus is not differentiable. Its construction begins with a shape such as the one shown in Figure 18a.* Next, each of the four line segments in this shape is replaced by a one–third scaled version of this same shape, as shown in Figure 18b. This recursive process is repeated infinitely to yield a Koch curve. The composition of three of these Koch curves creates a closed figure, often called a *Koch snowflake* (Figure 19). Because the Koch curve is exactly self–similar, a magnified view of any section of it will look the same as the original shape (although translation and rotation are allowed). This effect is illustrated in Figure 19.

Instead of replacing each line segment of the shape in Figure 18a with a scaled version of the same shape, suppose that different shapes may be used. As a simple example, suppose there are two shapes: the "normal" shape used in Figure 18a and an

---

* The Koch curves and Koch snowflakes illustrated here were created from an algorithm based upon a formal grammar model developed by Dietmar Saupe [38, Appendix C]. Each iteration applies a set of production rules to generate a character string which represents the curve at a given resolution. This string is then translated into a graphical portrait.

(a) First iteration   *(b) Second iteration*   *(c) Third iteration*

(d) After six iterations, no further detail can be discerned at this resolution
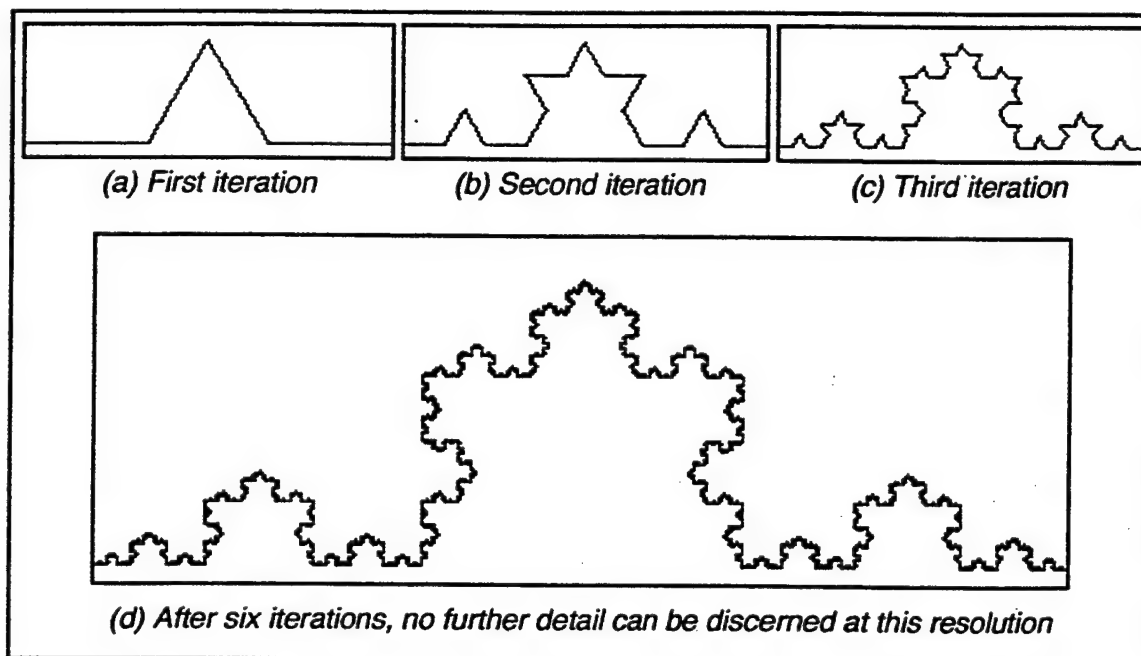
Figure 18 Construction of a Koch Curve
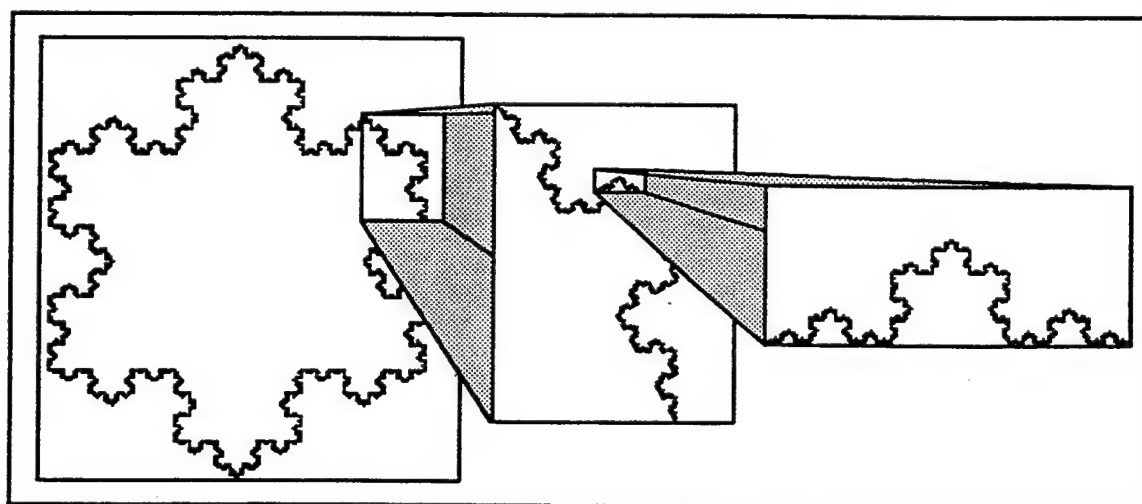
Figure 19 Successive Magnification of a Koch Snowflake

upside–down or inverted version of that same shape. The selection of which shape to use may be determined randomly. Two such random fractals are shown in Figure 20. Figure 20a shows the two shapes, normal and inverted. The fractal in Figure 20b contains mostly normal shapes (75%), and the fractal in Figure 20c contains an equal amount of normal and inverted shapes. (Note that the generation of these two fractals began with the same random seed value.) These two fractals are *statistically self-similar,*

(a) Two Shapes Used to Generate the Random Fractals

Normal

Inverted

(b) 75% Normal Shapes, 25% Inverted Shapes

(c) 50% Normal Shapes, 50% Inverted Shapes
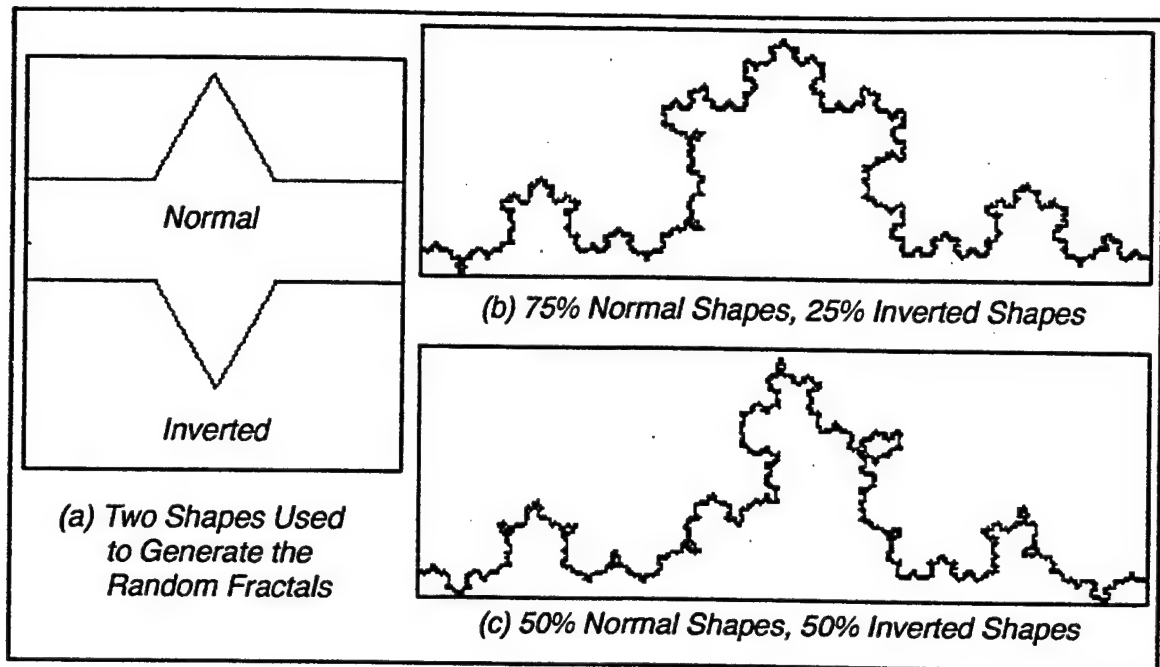
Figure 20 Statistically Self–Similar Koch Curves

since their self–similarity possesses randomness. One cannot determine exactly what shape should appear, but the overall process can be expected to follow certain statistical properties. Note that these statistically self–similar fractals contain an infinitely fine structure, just like their exactly self–similar counterpart in Figure 18d. Also, as will be demonstrated in section 3.4.2, the fractal dimensions of Figures 18d, 20b and 20c are all equal.

3.3.3 Examples of Self–Affine Fractals

In addition to rotation, reflection and translation, self–affinity allows each coordinate to be scaled by a different ratio. A matrix form is convenient for this representation, where $w(\mathbf{x})$ describes a geometric feedback equation often called an affine transformation.

$$\mathbf{x'} = w(\mathbf{x}) = \mathbf{Ax} + \mathbf{b}$$

Most applications involve two dimensions, with the customary notation shown below.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = w\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = \begin{bmatrix} a & b \\ c & d \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

A finite set of contractive affine transformations, called a Hutchinson operator [37], may be used to describe a particular self–similar or self–affine fractal. An iterated function system (IFS) [2] is the repeated feedback or iteration of a Hutchinson operator, which converges to a particular shape that defines the fractal. For example, the Koch curve may be expressed with four transformations, shown in tabular form in Figure 21. Let S be the set of points from $(0,0)$ to $(0,1)$, and let S_i be the set of points transformed by w_i, that is, $S_i = \{w_i(x) \mid x \in S\}$. The geometric interpretations of the first and second iterations of the Koch curve IFS are shown in Figures 22a and 22b, where S_iS_j indicates the set $\{w_i(w_j(x)) \mid x \in S\}$. (Some of the labels have been omitted in Figure 22b to reduce clutter.)

Barnsley's popular fern image [2] consists of the recursive composition of the four affine transformations shown in Figure 23. Starting with a unit square (i.e., with

	a	b	c	d	e	f
w1	0.3333	0.0000	0.0000	0.3333	0.0000	0.0000
w2	0.1667	-0.2887	0.2887	0.1667	0.3333	0.0000
w3	0.1667	0.2887	-0.2887	0.1667	0.5000	0.2887
w4	0.3333	0.0000	0.0000	0.3333	0.6667	0.0000

Figure 21 IFS Code for Koch Curve
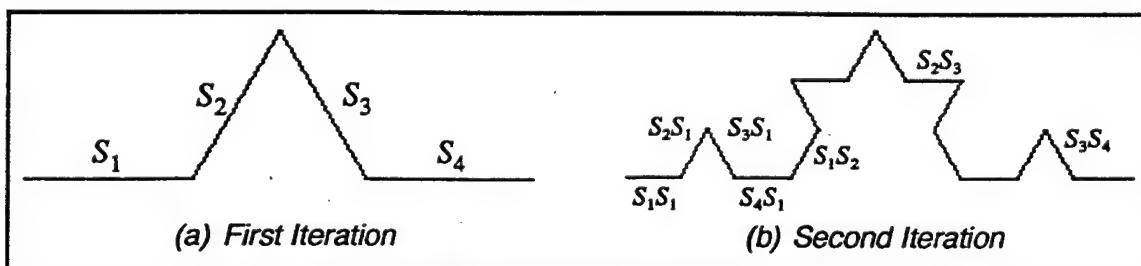
Figure 22 Construction of a Koch Curve Using an IFS

	a	b	c	d	e	f
w1	0.849	0.037	-0.037	0.849	0.075	0.183
w2	0.197	-0.226	0.226	0.197	0.400	0.049
w3	-0.150	0.283	0.260	0.237	0.575	-0.084
w4	0.000	0.000	0.000	0.160	0.500	0.000

Figure 23 IFS Code for Barnsley's Fern

coordinates $(0, 0)$, $(0, 1)$, $(1, 0)$ and $(1, 1)$), the first iteration is shown in Figure 24a. Eventually, successive iterations will surpass the displayed resolution, although the true fractal requires an infinite number of iterations. Note that the initial shape of the unit square is no longer apparent in the final image (Figure 24e). In fact, the same final result will be obtained with any initial shape, including even a single point.

Just as self–similarity may be exact or statistical, so can self–affinity. A random component in the selection of transformations is one way to obtain statistical self–affinity. Another type of statistical self–affinity is fractional Brownian motion, which will be described in the following section.

(a) One iteration (b) Two iterations

(c) Five iterations (d) Ten iterations (e) Enough iterations to exceed the resolution of the display
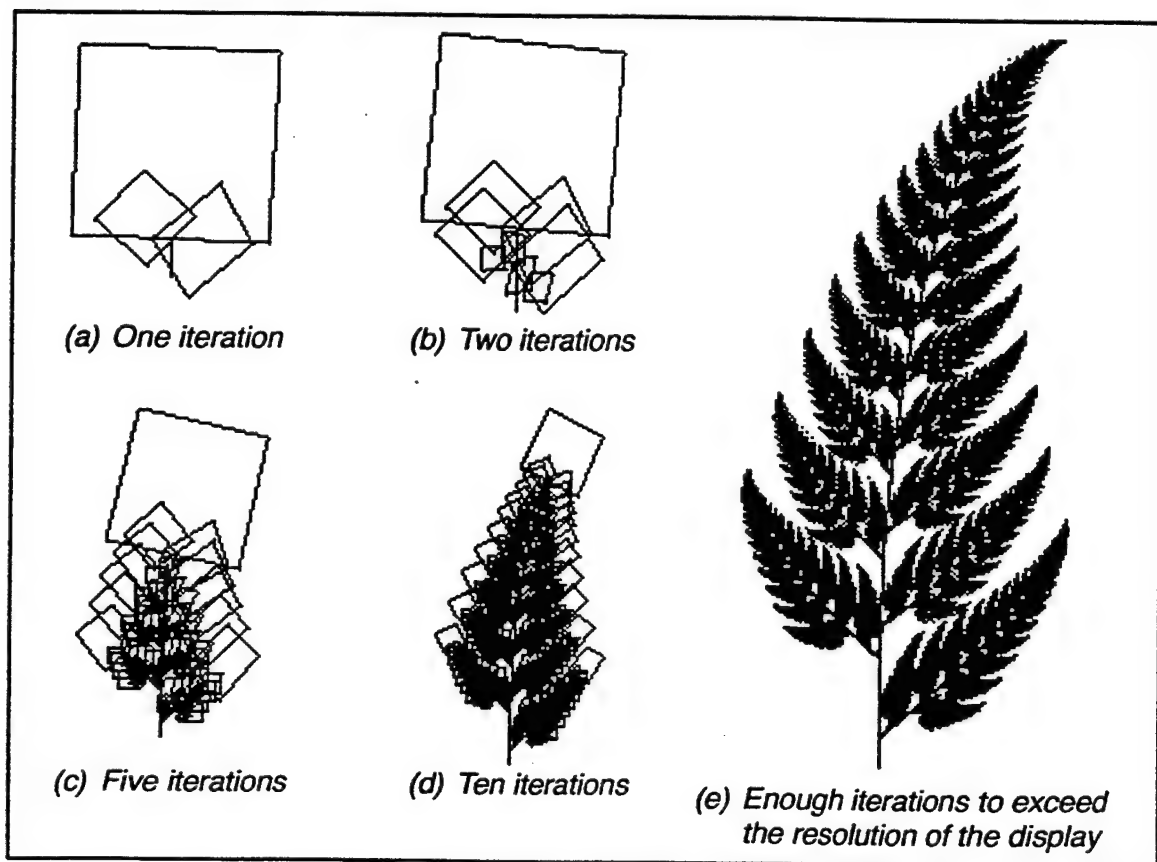
Figure 24 Construction of Barnsley's Fern Using an IFS

3.3.4 Fractional Brownian Motion

Introduced by Mandelbrot and Van Ness [34], fractional Brownian motion (abbreviated *fBm*) is a generalization of Brownian motion, which describes the complex, erratic movement of a particle, subjected to the collisions of the other particles in its surrounding medium [36]. The function of fBm is characterized by the properties of its increments, that is, the differences between successive samples. Let $B_H(t)$ represent a fBm signal, where t is a vector containing E independent variables.* Then $\Delta B_H = B_H(t_2) - B_H(t_1)$ specifies an increment of a fBm signal. The increments ΔB_H are normally distributed with a mean of zero and a variance proportional to the $2H$ power of the Euclidean distance between the two measurements t_1 and t_2. The mean and variance are expressed below, with σ^2 representing the proportionality constant of the variance. The Hurst parameter H must be between 0 and 1, exclusively ($0 < H < 1$).

$$E[\ B_H(t_2) - B_H(t_1)\ ]\ =\ 0$$

$$\text{var}[\ B_H(t_2) - B_H(t_1)\ ]\ =\ \sigma^2|\ t_2 - t_1\ |^{2H}, \qquad 0 < H < 1$$

(Informally, extensions beyond the range $0 < H < 1$ may be considered as "derivatives" and "integrals" of fBm [48].) Fractional Brownian motion becomes ordinary Brownian motion when $H = 1/2$. The value of H is related to the fractal dimension D by the formula $D = E + 1 - H$ (see section 3.4.5), where E is the Euclidean dimension (equivalently, the number of independent variables of the signal). Thus, small values of H yield a high fractal dimension, and large values of H produce a low fractal dimension.

Since $\text{var}[Y] = E[Y^2] - E[Y]^2$, the mean and variance equations above may be combined to give the expression

$$E[\ (B_H(t_2) - B_H(t_1))^2\ ]\ =\ \sigma^2|t_2 - t_1|^{2H}$$

---

* The notation used with fBm can vary considerably. When referring to a signal which is fBm *by definition*, the most common forms are $B_H(t)$ [31, 33], $B_H(u)$ [14] or $V_H(t)$ [47, 38]. However, frequently authors describe a signal which could be modeled or approximated by fBm, in which case there is no accepted format.

However, the more common form is below, which may be derived through a
transformation of random variables. (The constant k is proportional to the standard
deviation σ, as shown in the appendix.) From this equation

$$E[\ |B_H(t_2) - B_H(t_1)|\ ] = k\ |t_2 - t_1|^H$$

it is clear that scaling the input $\mathbf{t}$ by a constant r will result in the output being scaled by
r^H. This nonuniform scaling across all increment sizes makes fBm a particular instance
of a self–affine fractal.

Some examples of one–dimensional fBm are shown in Figure 25, with
dimensions $D=1.3$ ($H=0.7$), $D=1.5$ ($H=0.5$, which gives standard Brownian motion) and
$D=1.7$ ($H=0.3$). The time or $\mathbf{t}$ axis extends horizontally, with the function values $B_H(t)$
along the vertical axis. Note that the graphs have the same vertical range, as is indicated
by the thin horizontal lines across them.

As stipulated by the self–affine relationship, scaling the independent or $\mathbf{t}$ axis by a
constant r entails scaling the dependent or $B_H(t)$ axis by r^H. For example, a signal with
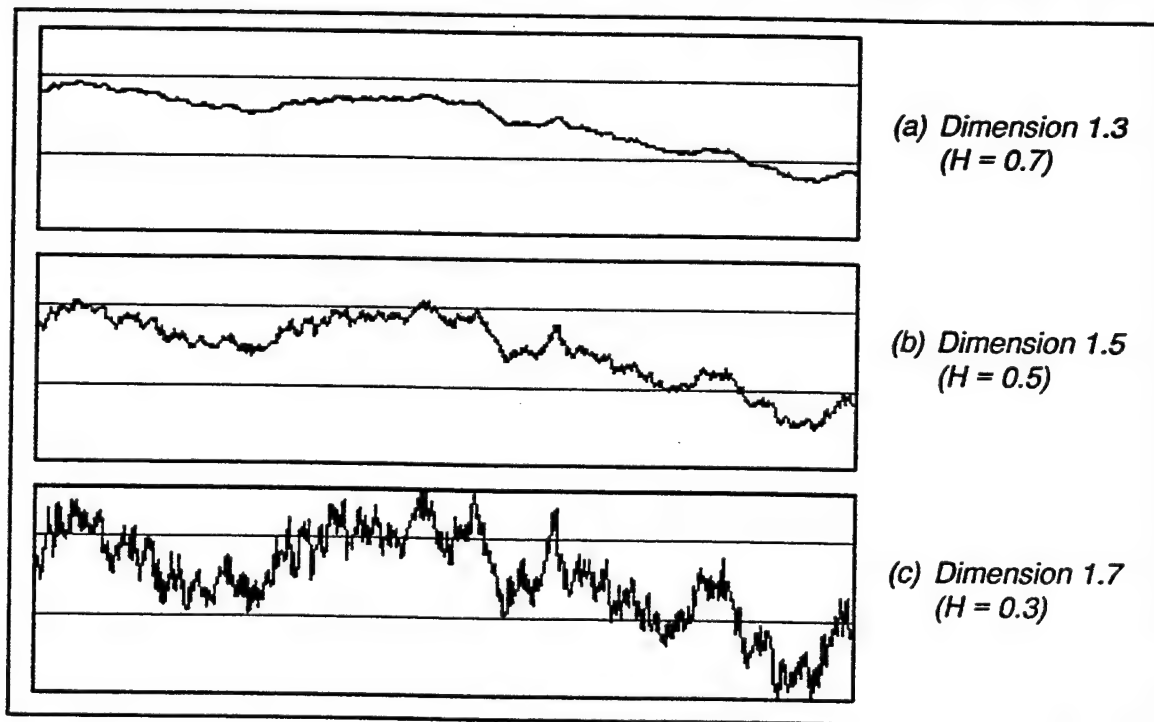
Figure 25 Examples of One-Dimensional Fractional Brownian Motion

dimension 1.7 is shown in Figure 26a. Figure 26b shows the first quarter of this signal, scaled improperly by the same factor of four along both axes. If the independent axis is scaled by four, then the dependent axis must be scaled by $4^{2-1.7} = 4^{0.3} \approx 1.5157$ for the signals to maintain identical statistical properties. The first quarter of the signal in Figure 26a is scaled properly in Figure 26c. Whereas the correctly scaled signal looks (and is) statistically identical to the original signal, the incorrectly scaled signal is clearly distinct. Note, however, that scaling the dependent axis alone will have a different effect. While the visual appearance will certainly differ, the fractal dimension will not be affected. A linear scaling of the dependent axis will change the proportionality constant, but not the H parameter, and therefore not the fractal dimension. Assumed or ignored in the literature, this fact allows the function values of the fBm signal to be normalized without altering the fractal dimension.

For a two–dimensional fBm signal, there are various ways to display the surface. One approach is to map each value to a certain color. For example, high values may appear light and low values may appear dark. This scheme is used in Figure 27, which shows three two–dimensional fBm surfaces, with fractal dimensions 2.3, 2.5 and 2.7. In contrast to the examples of one–dimensional fBm, the two–dimensional examples are scaled individually so that each signal extends from a minimum of 0 (black) to a
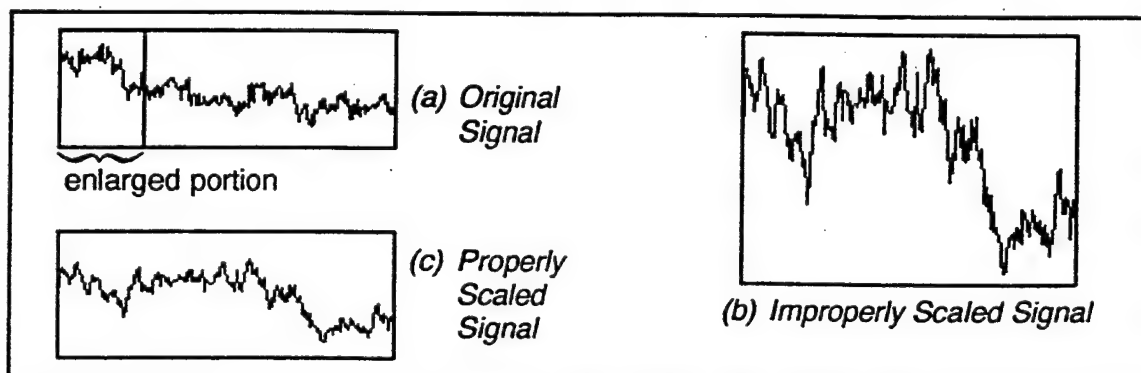
Figure 26 Scaling a Fractional Brownian Motion Signal

(a) Dimension 2.3 (b) Dimension 2.5 (c) Dimension 2.7
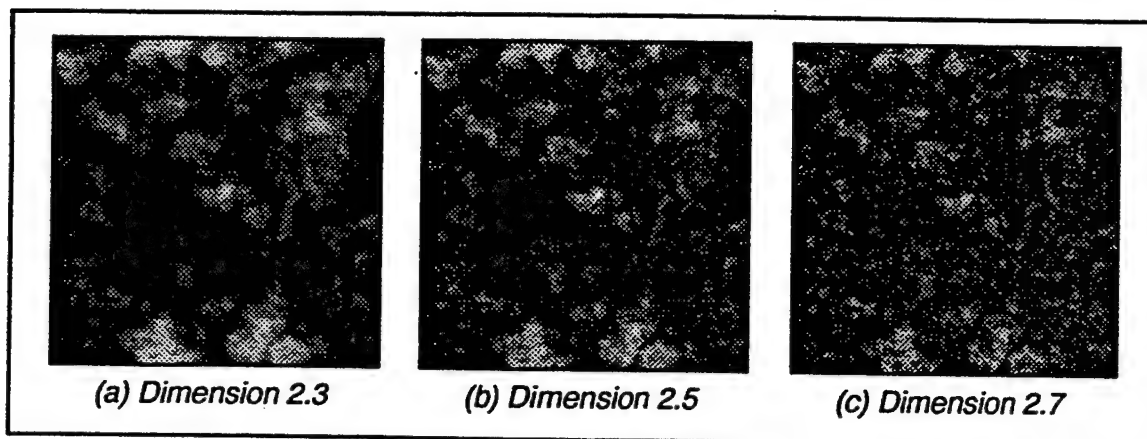
Figure 27 Examples of Two-Dimensional Fractional Brownian Motion

maximum of 255 (white). Likewise, fBm may be extended to higher dimensions, although it becomes more challenging to display the results.

3.4 Fractal Dimension

3.4.1 Introduction

Perhaps even more difficult than defining a fractal is defining the fractal dimension. In general, the fractal dimension attempts to quantify a subjective feeling about the degree to which a shape fills the Euclidean space in which it is embedded. As such, a fractional value is associated with the fractal dimension, rather than an integer. The fractal dimension provides a meaningful measure of fractal objects, both artificial and natural. For example, Mandelbrot [33] found that natural coastlines and terrain surfaces typically have fractal dimensions within the ranges 1.1–1.4 and 2.1–2.4, respectively.

Although intuitive motivation for fractal dimension is essential, formal support is also necessary. A plethora of formal definitions exist, many of which are very similar to one another, while others differ in significant ways. Mandelbrot [33], Falconer [13] and Peitgen et al. [37] provide excellent discussions of several different fractal dimension definitions, each reference with a distinct style and each including definitions that the others exclude. Falconer also outlines a typical set of desirable properties for a

dimension definition. Incidentally, he recognized the variety of existing dimension definitions in his fourth property describing a fractal (see section 3.2).

Although the definitions are all related, "some of them, however, make sense in certain situations, but not at all in others.... Sometimes they all make sense and are all the same. Sometimes several make sense but do not agree. The details can be confusing even for a research mathematician." [37, p.202] Experience and intuition guide the selection of an appropriate definition. Three of the most prominent definitions will be discussed here: the similarity dimension, the Hausdorff dimension and the box–counting dimension.

3.4.2 Similarity Dimension

One of the simplest definitions is the *similarity dimension,* although it applies only to fractals which are strictly self–similar. Nevertheless, it provides a clear intuitive link from Euclidean to fractal dimension. Initially, consider only Euclidean objects, such as a line, square or cube. Each of these shapes may be considered to consist of smaller copies of itself. For example, if a line is scaled by one third, then three such copies will produce the original line. Likewise, each of the three subsections may be similarly decomposed recursively into infinitely smaller, scaled line segments.

Clearly, scaling a line segment by $1/r$ will require r such copies to cover the original line. When a square is scaled by $1/r$, however, r^2 copies are needed to cover the original square. A cube requires r^3 copies of itself, each scaled by $1/r$. This effect is illustrated in Figure 28a for $r = 3$. The relationship between the scaling factor $1/r$ and the number of scaled copies N may be generalized to the expression $N(1/r)^{D_s} = 1$ or $N = r^{D_s}$, where D_S is the object's dimension. Alternately, if the dimension is unknown, it may be computed from the number of copies and scaling ratio:

$$D_S = \log(N)/\log(r)$$

Similar analysis may be conducted on self–similar fractals, such as the Cantor set (Figure 17) or Koch curve (Figure 18). However, whereas any scaling ratio was equally satisfactory in the previous example, the choice is restricted for most self–similar fractals. As outlined in its recursive construction in section 3.3.2, the ternary Cantor set consists of two copies of itself, each scaled by one third. (In general, there will be 2^n copies scaled by $1/3^n$.) Accordingly, its similarity dimension is $\log(2)/\log(3) \approx 0.6309$. The Koch curve contains four subcopies, each scaled by one third, yielding a similarity dimension of $\log(4)/\log(3) \approx 1.2619$. The geometric scaling relationships of the Cantor set and Koch curve are shown in Figure 28a. In Figure 28b the computation of the similarity dimension is shown for a few particular resolutions of a line, square, cube, Cantor set and Koch curve. Note that the similarity dimension is the same as the Euclidean dimension for the standard Euclidean objects.

shape	N	r	$D_S = \log(N)/\log(r)$
line	3	3	1
	10	10	1
square	9	3	2
	100	10	2
cube	27	3	3
	1000	10	3
Cantor set	2	3	0.6309
	4	9	0.6309
Koch curve	4	3	1.2619
	16	9	1.2619

N Number of parts or subcopies
$1/r$ Reduction factor for each subcopy
D_S Similarity dimension

(a) Representing a Shape as a Collection of Subcopies of Reduced Size

(b) Computation of Similarity Dimension
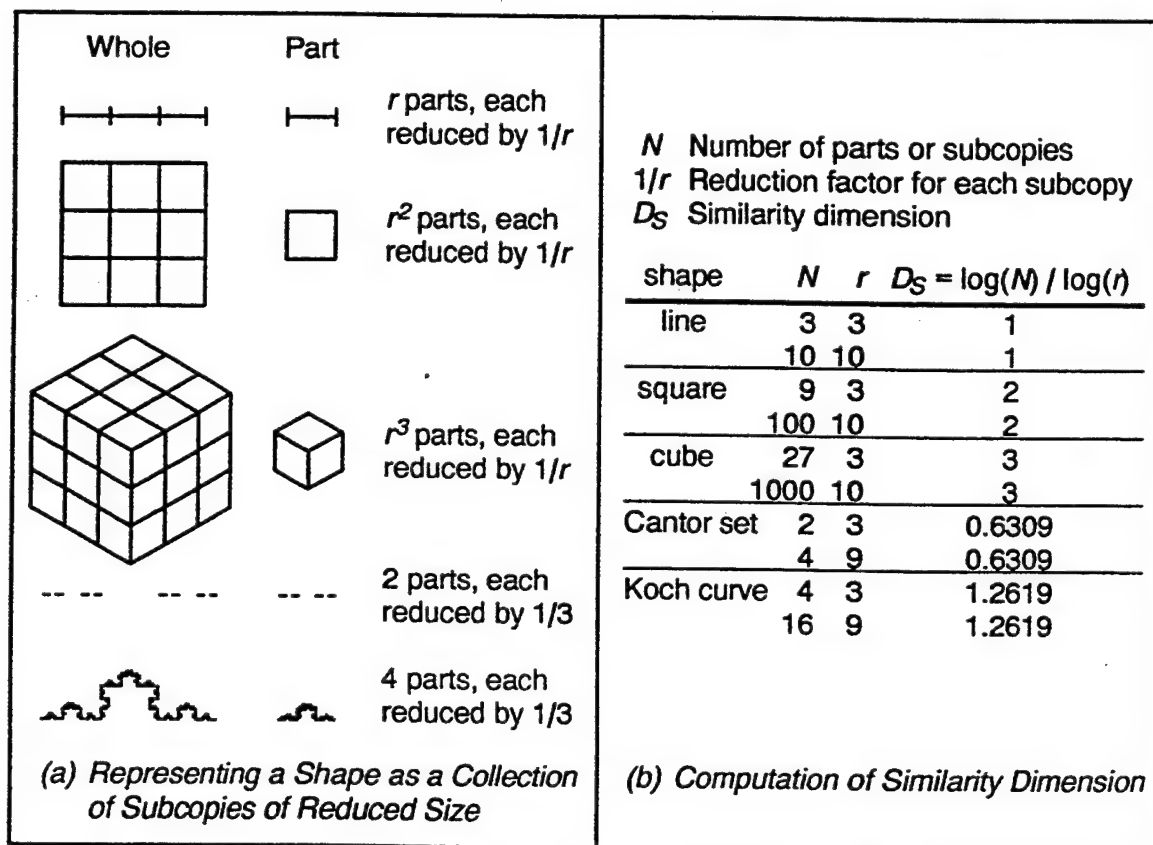
Figure 28 Illustration of the Similarity Dimension

3.4.3 Hausdorff Dimension

Although the simplicity of the similarity dimension is appealing, it is limited to only self–similar fractals. In contrast, the Hausdorff dimension (sometimes called the Hausdorff–Besicovitch dimension) is defined for any set. It is also the oldest definition, formulated by Felix Hausdorff in 1919 and is probably the most thoroughly studied. Thus, the Hausdorff dimension is often considered as the "default" fractal dimension, at least among mathematically oriented definitions.

Before encountering the mathematical theory, a pair of illustrative examples will be helpful for understanding the motivation behind the Hausdorff dimension. Length, area and volume are one, two and three dimensional measures, respectively. A square has infinite length, zero volume, but finite area, which corresponds to the fact that a square is a two–dimensional object. Consider the Koch curve (Figure 18), with a similarity dimension of approximately 1.2619. Its length, a one–dimensional measure, is infinite. (The length of the nth stage in the construction of the Koch curve is $(4/3)^n$, which becomes infinite as n approaches infinity.) Its area, a two–dimensional measure, is zero. As the following explanation will indicate, only at the fractional value $\log(4)/\log(3)$ will the Koch curve produce a finite measure.

Using the standard Euclidean distance $|x - y|$ between a pair of points x and y in n–dimensional Euclidean space, the diameter of a set U is defined as

$$\text{diam}(U) = \sup\{ |x - y| : x, y \in U\}$$

which is the greatest distance between any pair of points in U. The countable family of open sets $\{U_1, U_2, U_3, ...\}$ forms an open cover of set A if $A \subseteq \bigcup_{i=1}^{\infty} U_i$. An ε–cover of set A results when the sets U_i are all less than ε in diameter, for some $\varepsilon > 0$. For $s > 0$,

$$h_\varepsilon^s(A) = \inf\left\{ \sum_{i=1}^{\infty} \text{diam}(U_i)^s : \{U_i\} \text{ is a } \varepsilon\text{–cover of } A \right\}$$

The above equation takes the infimum across all open ε–covers of A. For each U_i in the cover, its diameter is raised to the sth power and summed together. The infimum of these summations may or may not be finite. The number of ε–covers decreases as ε approaches zero, leading to the s–dimensional Hausdorff measure of A:

$$h^s(A) = \lim_{\varepsilon \to 0} h^s_\varepsilon(A)$$

This limit exists for any set A in Euclidean space, although it is usually either zero or infinity. In fact, there is a critical value $D_H(A)$ such that

$$h^s(A) = \begin{cases} \infty & \text{if } s < D_H(A) \\ 0 & \text{if } s > D_H(A) \end{cases}$$

This value is defined as the Hausdorff or Hausdorff–Besicovitch dimension of set A.

$$D_H(A) = \inf\{s : h^s(A) = 0\} = \sup\{s : h^s(A) = \infty\}$$

A variety of fundamental properties of the Hausdorff dimension are discussed thoroughly in the second chapter of Falconer [13].

3.4.4 Box–Counting Dimension

In spite of the advantages of the Hausdorff dimension, it is usually difficult to calculate or estimate computationally. In practice, the box–counting dimension or a related variant serves as a useful alternative. Lending itself to straightforward empirical estimation, the box–counting dimension "is the one most used in measurements in all the sciences." [37, p.214] To some degree, the box–counting dimension also provides a link between the Hausdorff and similarity dimensions.

The difficulty of measuring the Hausdorff dimension lies in the evaluation of the summation of the $\text{diam}(U_i)^s$ terms (see section 3.4.3), since all $\text{diam}(U_i) < \varepsilon$ must be considered. The formula may be simplified by covering the fractal object with n–dimensional spheres or "balls" of fixed diameter ε. Or, one could use n–dimensional cubes or "boxes" of fixed length δ, which have a diameter $\varepsilon = \delta\sqrt{n}$. In this case, the

s–dimensional Hausdorff measure $h^s(A)$ is replaced by $N_\delta(A)\delta^s$ where $N_\delta(A)$ is the number of boxes of size δ covering the set A:

$$N_\delta(A)\delta^s = \lim_{\delta \to 0}\left\{\inf\left\{\sum_{i=1}^{\infty} \delta^s : \{U_i\} \text{ is a } \delta\text{–cover of } A\right\}\right\}$$

The value s for which $N_\delta(A)\delta^s$ is non–zero and finite becomes the box–counting dimension D_B. Roughly speaking, $N_\delta(A)\ \alpha\ \delta^{-D_B}$ for small s. (The proportionality may be replaced by an equality if δ specifies an absolute size instead of a relative size ratio.) The procedure may be simplified further by restricting the boxes to a mesh, eliminating the complication of overlapping and rotated boxes. Such a configuration may be implemented effectively on a computer.

The box–counting dimension may also be considered as a generalization of the similarity dimension (see section 3.4.2). The similarity dimension is $D_S = \log(N_r)/\log(r)$, but it applies only to self–similar fractals and usually requires a particular selection of r values. Let the box size $\delta = 1/r$. Then the box–counting dimension may be defined as

$$D_B = \lim_{\delta \to 0}\left\{\log(N_\delta)/\log(1/\delta)\right\}$$

which may be applied to any fractal. Typically, an algorithm will estimate D_B across a range of box sizes for better results. To compensate for using a fixed mesh of boxes, N_δ may be averaged across multiple meshes of size δ which are successively shifted.

3.4.5 The Dimension of Fractional Brownian Motion

The fractal dimension of fractional Brownian motion (fBm) may be estimated using the box–counting dimension described in the previous section. However, the relationship between the fractal dimension and the H parameter of fBm requires a distinction between two particular representations of fBm. Fractional Brownian motion can be graphed as either a *trail* or a *trace*. For example, Figure 29a shows a trace of
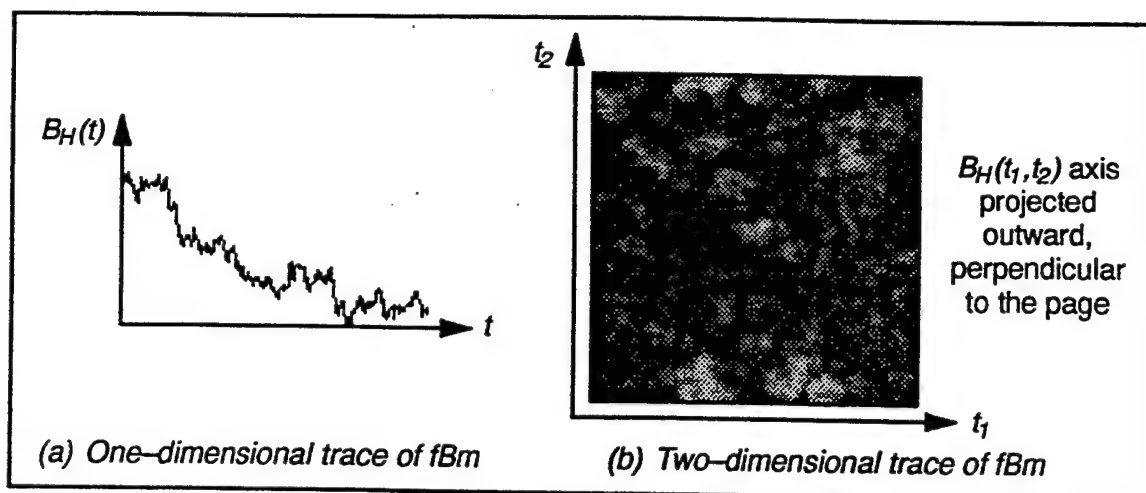
Figure 29 Traces of Fractional Brownian Motion

one–dimensional fBm. Note that the independent variable t follows one axis, and the function $B_H(t)$ follows the other axis. An overhead view of a trace of two–dimensional fBm is shown in Figure 29b, with the independent axes labeled t_1 and t_2 and the dependent axis $B_H(t_1, t_2)$ projected upward from the page. (The signal's height is indicated by the brightness.) In contrast, a trail of fBm removes the independent variable from the graph. Here, the path of the given particle is recorded by marking its position over time. The graph will contain a twisted line which meanders about, crossing over itself occasionally. The independent variable (time) is represented only implicitly. Finding the position of the particle at a particular time would require following along the trail a certain number of points. For a trace of fBm, however, the particle's position at a given time is readily apparent. Note that fBm trails will not be needed here; only fBm traces will be used.

The relationship between the fractal dimension D and the Hurst parameter H differs for fBm trails and traces.* Voss [48] determined that $D = 1/H$ for fBm trails, up to the dimension of the Euclidean space in which the signal is embedded. Thus, ordinary one–dimensional Brownian motion which fills the plane has $H = 1/2$ and $D = 2$. If

---

* Mandelbrot mentions the relationship between the Hurst parameter and fractal dimension as early as [30], but does not elaborate upon it. A decade later, Voss [48] provides a lucid explanation.

$1/H$ exceeds two, the trail overlaps itself more densely, but its dimension remains at two. The fractal dimension D of a fBm trace is $D = E + 1 - H$, provided that the function is considered over a sufficiently small scale of the independent variable. Note that, by convention, E refers to the Euclidean dimension of the signal, not the space in which the signal is embedded. More precisely, E is the number of independent variables of the signal. The following derivation, condensed from Voss [48], is included below, since this model is highly relevant to the fBm representation of image texture.

Recall from section 3.3.4 that the increments of fBm $\Delta B_H = B_H(t_2) - B_H(t_1)$ are, on average, proportional to the H power of the distance between the two measurements $\Delta t = t_2 - t_1$. Thus, on average, $\Delta B_H = k \, \Delta t^H$. A trace of a one–dimensional fBm signal is shown in Figure 30, sampled at intervals of δ. A square mesh with boxes of size δ has been laid atop this signal. Because the interval Δt equals the box size δ, $\Delta B_H = \delta^H$. (Note that the proportionality constant k has been dropped, for simplicity. The proportionality will not affect the derivation and will be re–introduced later.) The number of boxes needed to cover one time instance of this signal is the average height of the signal within the interval, divided by the box size. Thus, the number of boxes of size δ needed to cover a single vertical section of the signal is $N_{vertical}(\delta) = \Delta B_H / \delta = \delta^H / \delta = \delta^{H-1}$. The number of boxes horizontally is $N_{horizontal}(\delta) = \delta_{max} / \delta$, where δ_{max} is the smallest box size which covers the entire
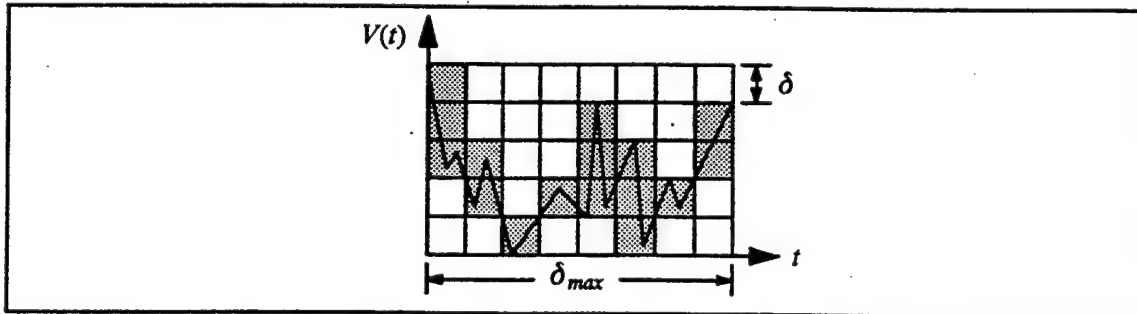
Figure 30 Box Counting Applied to Fractional Brownian Motion

function. The total number of boxes $N(\delta)$ needed to cover the entire signal is the product of the number of boxes horizontally times the number of boxes vertically:

$$N(\delta) = N_{horizontal}(\delta) \; N_{vertical}(\delta) = \frac{\delta_{max}}{\delta} \; \delta^{H-1} = \delta_{max} \; \delta^{H-2}$$

These boxes are shaded in Figure 30. Combining δ_{max} and the earlier constant k,

$$N(\delta) \; \alpha \; \delta^{H-2}$$

Recall from section 3.3.4 that

$$N(\delta) \; \alpha \; \delta^{-D}$$

Thus, $-D = H - 2$ or $D = 2 - H$. Voss [48] generalizes this relationship to higher dimensions, demonstrating that the fractal dimension of a fBm trace is $D = E + 1 - H$, where H is the Hurst parameter and E is the number of independent variables of the signal.

The spectral density $S(f)$ of a fBm function $B_H(t)$ also has special properties. In particular, the frequency distribution of fBm decreases exponentially as the frequency increases. Specifically, the spectral density is inversely proportional to the $\beta + E - 1$ power of the frequency magnitude $|\mathbf{f}| = \sqrt{\sum_{i=1}^{E}(f_i)^2}$, where $\mathbf{f}$ is a vector of size E and k is a constant:

$$S(f) = \frac{k}{|\mathbf{f}|^{\beta+E-1}}$$

The parameter β is related to H as shown below:

$$\beta = 2H + 1 \qquad H = (\beta - 1)/2$$

The derivation is excluded here; Peitgen and Saupe [38] provide further detail. From the above equation, the fractal dimension may be expressed in terms of β:

$$D = E + 1 - H = E + (3 - \beta)/2$$

Both the spatial and spectral forms of fBm will be used later in the analysis of image texture.

CHAPTER IV

FRACTALS AND TEXTURE

4.1 Modeling Natural Terrains and Textures with Fractals

Mandelbrot [30] noticed that natural terrain surfaces may be modeled by two–dimensional fractional Brownian motion with startling effectiveness. Introduced by Mandelbrot and Van Ness [34], fractional Brownian motion (often abbreviated *fBm*) is a generalization of Brownian motion, which describes the irregular path of a particle in a fluid, caused by the random molecular impacts of the surrounding particles. Voss [47] established a link between the spectral distribution of fBm and the fractal dimension. Based on this, he used Fourier synthesis to generate natural terrains and even complete landscapes. Upon closer examination, however, a simple fBm surface cannot accommodate the full diversity of detail present in an actual landscape. For example, fBm does not accurately portray lake bottoms [33] (which are submerged and thus invisible in the artificial landscapes), and river networks are generally absent in typical fractal renderings. However, this does not invalidate the suitability of fractals for modeling natural terrain. It merely indicates that there may be a combination of fractal phenomena involved. For example, a fractal approach was used in [38] to include river networks within a fBm landscape. Pentland [39] supports such a view, arguing that natural terrain should follow a fractal model, since it is formed by fractal processes such as aggregation and erosion. Furthermore, he asserted that an image intensity surface of a fBm surface will also produce fBm, under normal lighting conditions. In other words, not only may terrain be modeled by fBm, but images of such terrain also.

Recognizing that fBm may be used to study images of natural terrain, Pentland [39] extended fBm to image textures in general. He found that fBm relief maps of

varying fractal dimension correlated very closely with the human perceptual notion of roughness, a fundamental characteristic of texture. Estimating the fractal dimension from the Fourier transforms of local neighborhoods, Pentland distinguished the textures within a sampling of Brodatz textures and textures from natural scenes. The estimated fractal dimension was also used to segment the regions within images of a desert, San Francisco Bay and Mount Dawn. Keller et al. [24] developed a one–dimensional fBm model to distinguish between silhouettes of trees and mountains.

Natural image texture models are not restricted to using only the fractal dimension. In later work, Keller et al. [25] supplemented the fractal dimension with a lacunarity estimate to segment a set of Brodatz textures and a scene of trees. Likewise, Vehel [46] implemented an alternate form of each lacunarity and the fractal dimension to classify images of human lungs. Generalizing the normally scalar fractal dimension, Kaneko [23] used a 2×2 fractal dimension matrix to capture directional dependencies. The eigenvalue of this matrix was used to characterize Brodatz textures. Dennis and Dessipris [11] studied the derivative (rate of change) of the fractal dimension as a texture signature across a range of pixel distances. Wanting to distinguish between natural and cultural (man–made) objects in aerial images, Cooper and Chenoweth [7] investigated "fractalness," or the degree to which an image region fits the fractal model.

Here, two primary fractal characteristics will be explored: fractal dimension and "fractalness." This work is oriented toward the representation of natural textures in aerial imagery, although Brodatz textures and artificial fractal textures are also included. The development of a fractal model for visual texture representation is divided into two parts: the synthesis of fractal images and the estimation of the fractal properties of images. The remainder of this chapter focuses on the generation of artificial images to be used for testing. The next chapter describes various approaches to extracting fractal characteristics of both artificial and actual images.

4.2 Synthesizing Artificial Fractal Images

Before developing a method for estimating the fractal dimension of an actual image or of subregions of an actual image, a test standard is needed by which the estimation technique may be judged. The test standard should be created in a controlled manner such that its textural description, in this case, fractal dimension, originates from a sound theoretical model. Two theoretical approaches were examined and implemented for synthesizing artificial fractal images, a spectral technique (section 4.3) and the random midpoint displacement method (section 4.4).

Fractional Brownian motion may be characterized by the exponential relationship below (see section 3.3.4), where the Hurst parameter H satisfies $0 < H < 1$ and k is a positive constant:

$$E[\ |B_H(t_2) - B_H(t_1)|\ ] = k\ |t_2 - t_1|^H$$

For traces or functions of fBm (see section 3.4.5), the Hurst parameter is related to the fractal dimension D by the equation $D = E + 1 - H$, where E is the Euclidean dimension of the signal, which is the number of independent variables. Small values of H yield a rough visual texture; large values of H produce a smoother texture. Alternately, fBm may be expressed by its frequency distribution. As was shown in section 3.4.5, the spectral density $S(f)$ of fBm is inversely proportional to the $\beta + E - 1$ power of the frequency magnitude $|f|$, where f is a vector of size E (k is a constant):

$$S(f) = \frac{k}{|f|^{\beta + E - 1}}$$

The spectral parameter β is related to the Hurst parameter H by the equation $\beta = 2H + 1$. Thus, the fractal dimension may also be expressed in terms of β as $D = E + (3 - \beta)/2$.

When considering a two–dimensional signal, such as an image, there will be two independent variables ($E = 2$). The notation $G[x]$ refers to such a signal, which will be compared to the fBm signal $B_H(t)$. Thus, $x = (x_r, x_c)$, which specifies the discrete row and column coordinates. The function $G[x] \in [0, ..., 255]$ is a discrete value

representing the gray shade at location **x**. The following sections describe algorithms to synthesize a fBm image $G[\mathbf{x}]$, first in the frequency domain and then in the spatial domain.

4.3 Spectral Synthesis

4.3.1 Algorithm Design

For a two–dimensional signal ($E = 2$), the spectral density of fractional Brownian motion is

$$S(f) = S(f_1, f_2) = \frac{k}{(\sqrt{f_1^2 + f_2^2})^{\beta + E - 1}} = \frac{k}{(\sqrt{f_1^2 + f_2^2})^{\beta + 1}} = \frac{k}{(f_1^2 + f_2^2)^{\frac{\beta+1}{2}}}$$

or, in terms of the Hurst parameter H,

$$S(f_1, f_2) = \frac{k}{(f_1^2 + f_2^2)^{H+1}}$$

In terms of the coefficients $V[n_1, n_2]$ of the two–dimensional discrete Fourier transform,

$$E[\ |V[n_1, n_2]|^2\ ] = \frac{k}{(n_1^2 + n_2^2)^{H+1}}$$

(where $E[Y]$ is the expected value of Y), which may be achieved by selecting coefficients such that

$$|V[n_1, n_2]| = \frac{1}{(n_1^2 + n_2^2)^{\frac{H+1}{2}}} = (n_1^2 + n_2^2)^{-\frac{H+1}{2}}$$

The spatial signal corresponding to the inverse Fourier transform of this frequency signal will be fractal, with dimension $3 - H$.

The program used to generate an artificial fractal image was based upon Saupe's algorithm [38] to synthesize two–dimensional fractional Brownian motion spectrally. As described previously, the synthesis begins in the complex frequency domain. The two–dimensional signal is conjugate symmetric and periodic. The lowest (zero) frequency component is located at the upper left corner of the array. Frequencies increase toward the center of the array, at which point they decline again, due to the periodic nature of the signal (i.e., the origin of the frequency plane has been shifted to the

upper left corner of the array). This relationship is shown in Figure 31, in which the array indices range from 0 to 255 and the frequencies n_1 and n_2 increase from 0 to a peak of 128 and then decline back to 1. The arrows indicate the direction of increasing frequency.

To assure that the spatial transform will be purely real (i.e., no imaginary components present), the frequency domain signal must be conjugate symmetric:

$$S(n_1, n_2) = S(-n_1, -n_2)*$$

Since the frequency domain signal is periodic, this becomes

$$S(n_1, n_2) = S(N - n_1, N - n_2)*$$

where N is the size of the array. Conjugate symmetry was included by setting the values in the lower right quadrant to the complex conjugates of the corresponding values in the upper left quadrant. Likewise, the upper right and lower left quadrants were linked by conjugate symmetry.

To give the final spatial image a more natural appearance, randomness was added to the frequency coefficients. Following Saupe's specifications [38], the phase of the coefficients was uniformly distributed between 0 and 2π radians using a built–in pseudorandom number generator. The magnitude of the coefficients was scaled by a Gaussian distribution with zero mean and unit variance. The design of the Gaussian
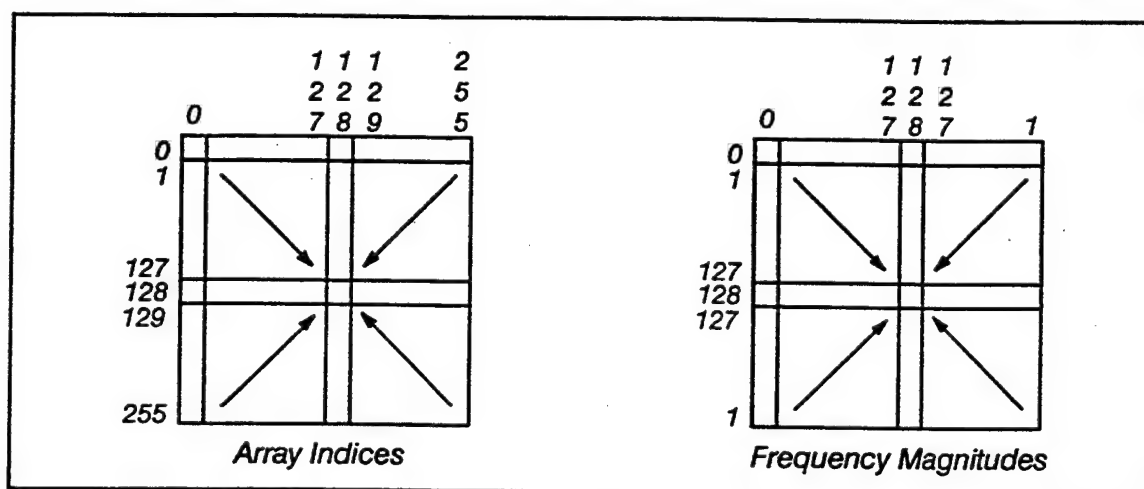
Figure 31 Spectral Synthesis: Array Indices and Frequency Magnitudes

pseudorandom number function capitalized upon the Central Limit Theorem, in which the mean of the sample of any distribution (in this case, the existing uniform generator) will approximate a Gaussian distribution. Justification of these random elements was absent in the original reference, and no other sources were located which address the matter. However, the following explanation is offered. First, the phase is unrestricted, so any random distribution should be acceptable. A uniform distribution across all phase angle values would seem reasonable. Justification of the magnitude is less direct. The magnitude must satisfy the relationship given below:

$$E[\ |V[n_1, n_2]|^2\ ] = \frac{k}{(n_1^2 + n_2^2)^{H+1}}$$

Using elementary probability theory, this may be rewritten

$$E[\ (V[n_1, n_2])^2\ ] = var[V[n_1, n_2]] + E[V[n_1, n_2]]^2$$

The fBm spectral requirements may be satisfied by selecting a random distribution with a zero mean and variance proportional to $(n_1^2 + n_2^2)^{-(H+1)}$. Multiplying a standard Gaussian distribution (i.e., with zero mean and unit variance) by $(n_1^2 + n_2^2)^{-(H+1)}$ achieved this specification.

Finally, the randomized spectral image was transformed to the spatial domain using an inverse fast Fourier transform. As explained, the spatial image will have no imaginary component, since the spectral image was conjugate symmetric. The magnitude of the real portion of the spatial image was normalized to the maximum range of displayable gray shades, from 0 to 255, and quantized to an integer value. A fractal signal may be normalized in a linear fashion without affecting its fractal dimension. (This property was demonstrated both analytically (see section 3.3.4) and through examination of the empirical results.) Three samples of synthetic fractal images generated in this manner are shown below in Figure 32. The fractal dimension D is related to the parameter H by the equation $D = 3 - H$ (assuming a two–dimensional signal). Notice how the visual texture becomes increasingly rough as the fractal dimension increases.

(a) *H = 0.9*
Fractal dimension = 2.1

(b) *H = 0.5*
Fractal dimension = 2.5

(c) *H = 0.1*
Fractal dimension = 2.9
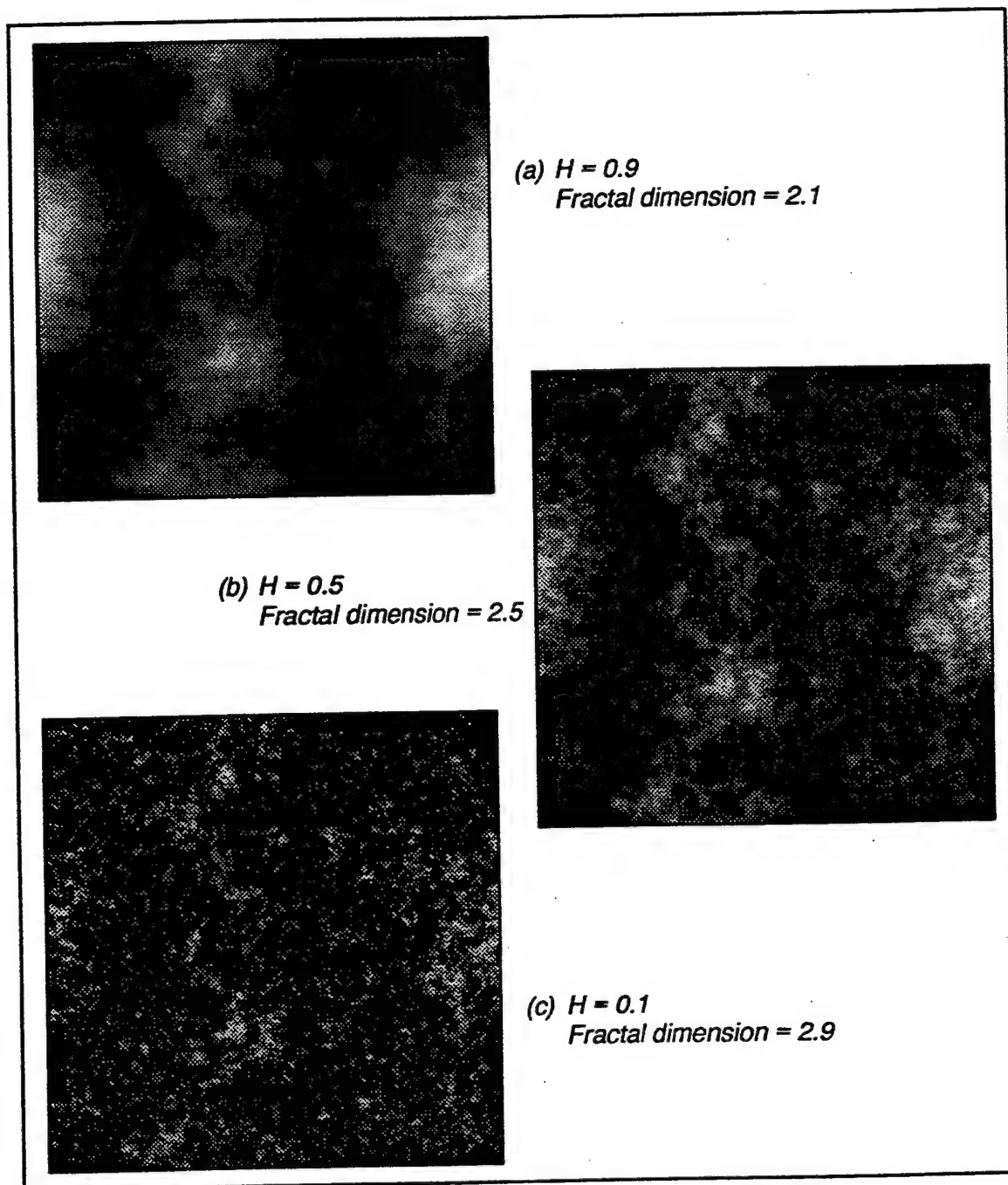
Figure 32 Synthetic Fractal Images, from the Spectral Model

4.3.2 Analysis of Results

The visual appearance of the synthetic fractal images seemed favorable. Cross

sectional slices of the images were taken and studied. Again, the results appeared

reasonable. Slices of several images were obtained, at the same location within each image. Similarities were apparent, due to the fact that each of the images used the same initial value for the pseudorandom number generation. However, a rougher signal was observed in direct relationship to the magnitude of the fractal dimension. Of additional significance was the fact that the minimum, maximum and average gray shades for all of the synthetic fractal images were identical. Thus, any method which attempts to distinguish such images must rely solely upon textural properties. Since the average and extrema gray shades were the same for each of the synthetic images, these factors could not inadvertently influence the identification.

In spite of their visual appeal, when the synthetic images were examined analytically, the results were less favorable. A linear regression approach within the frequency domain (see section 5.4.1) was used to estimate the fractal dimension. After establishing the validity of the linear regression algorithm, the inaccuracies in the estimates of the fractal dimension in the final images led to an examination of the algorithm which generates the original spectral image. Figure 33 lists the results for five synthetic fractal images with dimensions from 2.1 to 2.9. The leftmost column ("actual D") indicates the fractal dimension parameter which was used to generate the image. The fractal dimension was estimated from the spectral image as it was generated. Initial

actual D	estm. D_1	estm. D_2	estm. D_3
2.1	2.0992	2.1048	2.1493
2.3	2.2996	2.3042	2.3264
2.5	2.4999	2.5048	2.5480
2.7	2.6999	2.7050	2.7878
2.9	2.8996	2.9051	3.0565

actual D . . . . ideal fractal dimension used to generate the image
estimated D_1 ... estimate from the original spectral image, uncorrupted by noise
estimated D_2 ... estimate from the original spectral image, corrupted by noise
estimated D_3 ... estimate from the final quantized spatial image, corrupted by noise

Figure 33 Estimates of the Fractal Dimension of Spectrally Synthesized
Fractal Images

measurement errors led to the removal of the pseudorandom number functions. The second column ("estm D_1") gives the estimated dimension, measured directly from the spectral image, without any noise. Subsequent testing of the pseudorandom number functions revealed acceptable performance of both the uniform and normal distributions. A refinement of the estimation algorithm produced reasonable values (see the third column, "estm D_2," of Figure 33) for the fractal dimension of the spectral image, with noise added to the signal as described previously. However, a degree of error remained in the estimates obtained from the final synthetic image. These results are listed in the fourth column ("estm D_3") of Figure 33. Here, the spectrally synthesized image was transformed to the spatial domain, quantized and written to a file. Examples of such images were shown in Figure 32. These images were then converted back to the frequency domain, where the fractal dimension was then estimated. Because the synthesis and estimation algorithms were directly related, one should expect very accurate results. Accuracy losses attributed to computational error, such as converting between the spatial and spectral domains, should be negligible. Increased precision did not significantly improve the estimates, confirming this supposition. The remaining factor is the quantization of the real–valued signal into only 256 gray shade values. Further investigation is warranted to confirm that quantization is responsible for the discrepancy between the original dimension and the estimated dimension obtained from the final normalized, quantized spatial image. Nevertheless, in spite of this difference, the estimates are monotonically related to the actual values and fairly close for smaller dimensions. Additionally, the visual appearance of a series of synthetic images with increasing fractal dimension corresponds well with one's expectation of increasing textural roughness.

4.4 Random Midpoint Displacement Synthesis

4.4.1 Synthesis of Fractional Brownian Motion

The second theoretical model investigated for creating an artificial fractal image is the random midpoint displacement approach. This method follows directly from the definition of fractional Brownian motion given previously and repeated below.

$$E[\ B_H(t_2) - B_H(t_1)\ ] = 0$$

$$\text{var}[\ B_H(t_2) - B_H(t_1)\ ] = \sigma^2 |\ t_2 - t_1\ |^{2H}, \qquad 0 < H < 1$$

As before, the signal $B_H(t)$ may be represented by the discrete function $G[x]$, where $G[x]$ is the gray shade at the row and column coordinate specified by $x = (x_r, x_c)$.

Let the coordinate system be normalized so that its four corners have coordinates (0,0), (0,1), (1,0) and (1,1), whose gray shades are obtained randomly from a normal distribution with zero mean and variance σ^2. These four initial points are labeled with zeroes in Figure 34. The midpoint between these four points is a single point at the center of the grid, at a distance of $1/\sqrt{2}$ from each of the corners. This point, from the first midpoint displacement, is labeled with a one in Figure 34. Its gray shade is the average of its four neighbors', plus a random offset with zero mean and variance $(1/\sqrt{2})^{2H}\sigma^2$. This assures that the fractional Brownian motion requirements given above
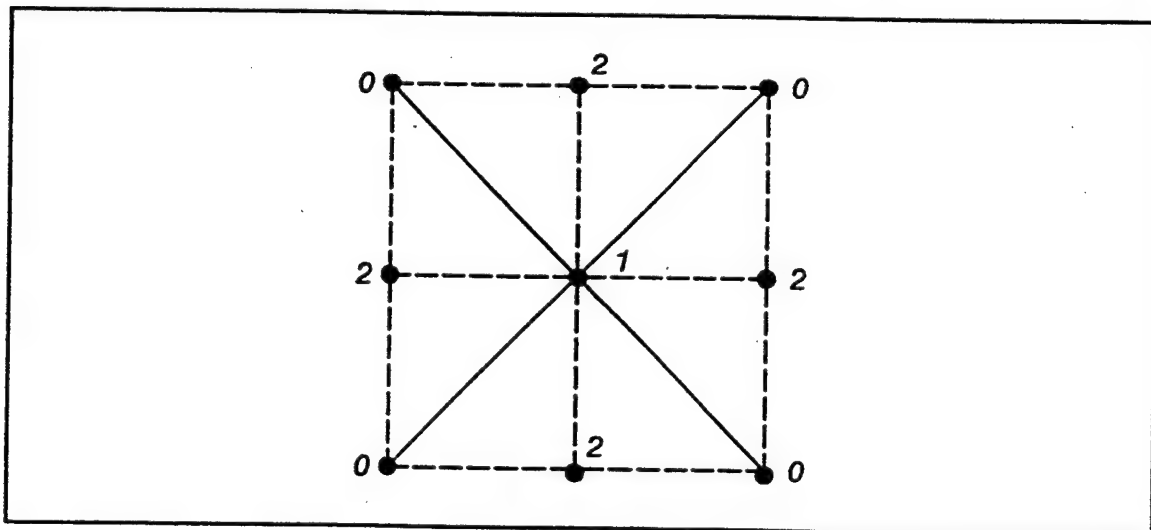
Figure 34 Initial Steps of Random Midpoint Displacement

are met. In the second midpoint displacement, the average of the gray shades of each pair of adjacent corners plus the center point is found, added with a random offset with zero mean and variance $(1/2)^{2H}\sigma^2$. This progression continues, alternating between adding points on the diagonally oriented grid (aligned with the solid lines of Figure 34) and adding points on the standard grid (aligned with the dashed lines of Figure 34). The final result is shown in Figure 35 for three different fractal dimensions.

Thus, the properties of fractional Brownian motion are preserved between the new point and those points whose values were included in the average. However, the same cannot be said about the relationship between points such as $(0, 1/4)$ and $(0, 3/4)$ which were generated independently from one another. Since the distance between points $(0, 1/4)$ and $(0, 3/4)$ is $1/2$, then the variance of the difference between their gray shades should be $(1/2)^{2H}\sigma^2$, in the same manner that the variance of the difference between the gray shades at $(0, 0)$ and $(0, 1/2)$, for example, is $(1/2)^{2H}\sigma^2$. However, this will be true only when $H = 1/2$. The actual variance between separately generated points will be too large for $H < 1/2$ and too small for $H > 1/2$. For other independently generated points, the theoretical validity fails even for $H = 1/2$. Thus, although the random midpoint displacement algorithm produces valid fBm to some degree, it falls short from a complete representation.

A variation of the random midpoint displacement algorithm adds a suitable displacement not just to the newly interpolated points, but to the original points as well. This process is called successive random addition [38]. Three samples of synthetic fractal images are shown in Figure 36, generated by the random midpoint displacement algorithm with random additions. This approach was motivated by the desire to eliminate the "creasing effect" which occurs in two-dimensional random midpoint displacement surfaces, when rendered as an illuminated terrain surface and viewed from certain angles, such as overhead (zenith) or parallel to the alignment of the square grid used by the algorithm. First mentioned by Mandelbrot [32], the creasing effect is

(a) *H = 0.9*
Fractal dimension = 2.1

(b) *H = 0.5*
Fractal dimension = 2.5

(c) *H = 0.1*
Fractal dimension = 2.9
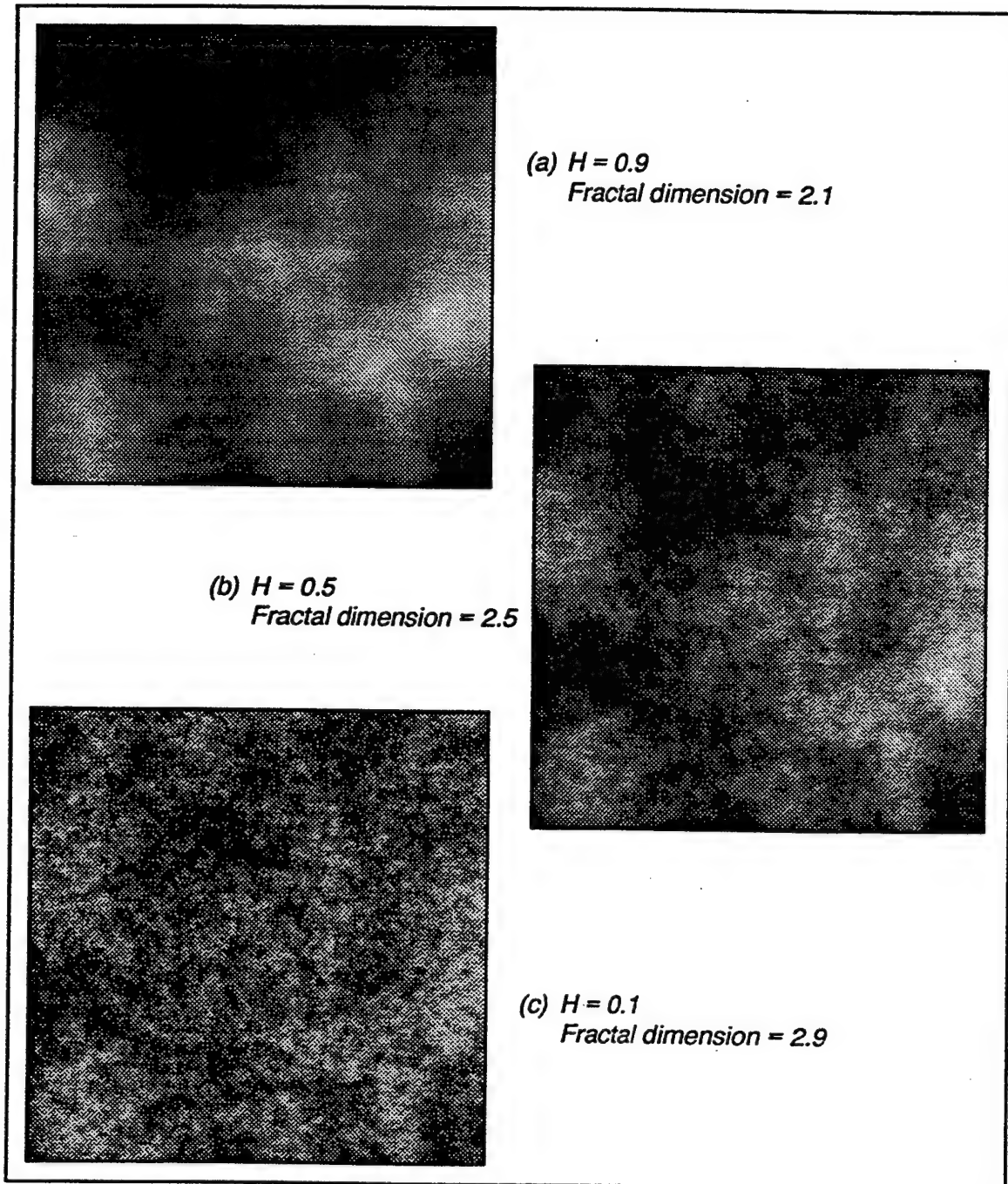
Figure 35 Synthetic Fractal Images, from the Random Midpoint Displacement Model

characterized by unnaturally long and straight features on the surface which resemble crumbled or creased paper.

Note, however, that the creasing effect is not visible when the surface is displayed as a "cloud," that is, when the height of the surface is mapped to a gray shade. For

(a) *H = 0.9*
Fractal dimension = 2.1

(b) *H = 0.5*
Fractal dimension = 2.5

(c) *H = 0.1*
Fractal dimension = 2.9
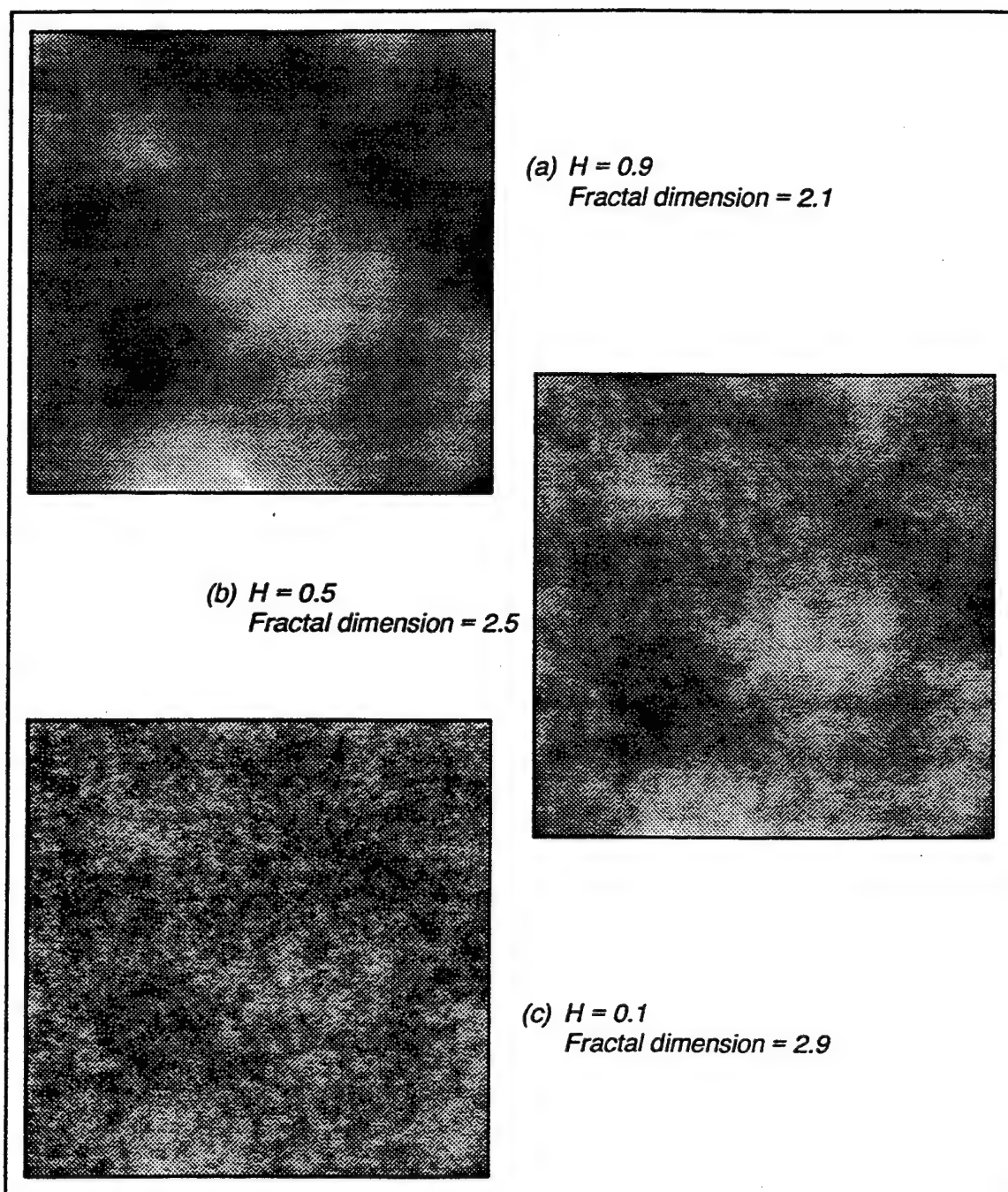
Figure 36 Synthetic Fractal Images, from the Random Midpoint Displacement Model
with Random Additions

example, refer to the fractal images in Figure 35. Likewise, even for illuminated terrain

surfaces, the proper selection of viewing angle will obscure the creasing effect.

Nevertheless, inclusion of random additions does not appear to affect the visual

roughness of the texture. For example, the three fractal images in Figure 36, which were created with random additions, compare very closely to their counterparts in Figure 35.

The successive random addition method includes a random offset not just to the newly generated midpoints, but to all of the points which have been created thus far in the algorithm. Although this eliminates the visual defect of creasing, it actually worsens the theoretical validity of the final signal. It was discovered that proper fBm is obtained only in the iteration in which a midpoint was newly created. The successive random additions of later iterations alter the variance even for those increments which were valid under the original random midpoint displacement algorithm.

4.4.2 Analysis of Results

The visual roughness of textures generated both with and without successive random additions appeared suitable. However, when the fractal dimension was estimated for these images, poor results were obtained. The first column ("actual D") of Figure 37 indicates the fractal dimension parameter used to generate the synthetic images. The second column ("estm D_1") contains estimates from images computed without successive random additions, and the third column ("estm D_2") shows the estimates taken from images which included successive random additions. The estimates were obtained by

actual D	estm. D_1	estm. D_2
2.1	2.5683	2.5299
2.3	2.7359	2.7016
2.5	2.9004	2.8723
2.7	3.0469	3.0310
2.9	3.1791	3.1685

actual D . . . . ideal fractal dimension used to generate the image
estimated D_1 . . . estimate from the final quantized image, without random additions
estimated D_2 . . . estimate from the final quantized image, with random additions

Figure 37 Estimates of the Dimension of Random Midpoint Displacement Fractal Images

transforming the fractal image into the frequency domain and then using linear regression (see section 5.4.1) on the spectral image. While such an approach should work well with the spectral synthesis algorithm, since both the synthesis and estimation paradigms are the same, it is conceivable that the spectral estimation of the random midpoint displacement algorithm could result in less accuracy. In contrast, values obtained from spatial estimation algorithms (see section 5.4.2) were much more reasonable. Incidentally, the apparent similarity between the visual roughness of textures generated with and without random additions is supported by the spectral estimates. In spite of the discrepancy between the actual and estimated dimensions, the estimated values corresponded closely between the two synthesis algorithms.

4.5 Summary

Of the three synthesis algorithms implemented, only the spectral approach was theoretically valid. Presumably, this is the reason why it is used exclusively in the literature. The other two types of synthesis, both based on random midpoint displacement, were found to have flaws which prevented them from adhering completely to the fBm requirements. No known studies have included the random midpoint displacement method in a rigorous analysis. (Only terrain simulations have used this algorithm, where the judgment considered only the subjective appearance of the rendered surface.) In spite of the theoretical shortcomings of the spatial algorithms, they were included with the spectral synthesis technique for estimation in the following chapter, to provide a more comprehensive comparison.

CHAPTER V

ESTIMATION RESULTS

5.1 Using Texture to Segment Aerial Imagery

In previous work contracted by the Naval Air Warfare Center in Indianapolis, Indiana, texture features were examined to aid the segmentation of aerial imagery [6]. The natural textures which appear in aerial images are frequently composed of very small texture elements, often considered to be the individual pixels which compose the image. Their arrangement tends to be statistical and non–periodic. Statistical and discrete transform methods are suitable for such textures. Of this category, techniques examined in the study included a small set of ad hoc filters, busyness and texture energy. Also considered was the gray shade co-occurrence matrix method, which is appropriate for pixel–sized texture elements with a statistical distribution. Although it is designed for much larger texture elements, the chord transform was evaluated as a potential tool for analyzing the shapes of certain regions.

The texture study began with two sets of simple, arbitrary masks, one which considers the directional content of the texture and the other which looks for checkerboard patterns. This approach required the selection of an exemplar texture to which the rest of the image would be compared. Since results can vary appreciably according to the exemplar texture, this requirement is generally a disadvantage. Also, the ad hoc masks tended to overemphasize the gray shade, and sometimes edge, content of the image. Conceivably the ad hoc masks could be supplemented by subsequent processing which would collect local statistics from the filtered images, as Laws did in his texture energy model [28]. However, this concept was not investigated.

The busyness feature [43] is a quick, non–linear filter which considers the average absolute gray shade differences within a small neighborhood of pixels. Since this approach can also extract edge information, the minimum of these averages was taken from four directions. This suppresses large gray shade differences resulting from edges. Results were favorable for both a synthetic test image and a set of aerial images. The only problem was that sometimes edges at angles not measured by the busyness operator would be marked with high busyness values.

Lastly, the texture energy model originated from Laws' work [28] was examined. Although certain features of its final fifteen "texture energy planes" rivaled busyness, no single particular feature was consistently preferred. It is conceivable that the complete texture energy model, when coupled with gray shade information, could give comparable or superior results to the combination of busyness and gray shade. However, the extra benefit anticipated would not justify the additional computational expense.

Thus, this examination of texture features led to the selection of busyness, which performed well with the natural textures of aerial images and was fast to compute. Busyness was used to model texture within the Gray Shade and Texture Segmentation Software, a tool designed to provide the scene analyst with a way to achieve improved accuracy in the segmentation of aerial images [5]. The combination of busyness and gray shade allowed much better classification of the regions within aerial images, as demonstrated by the results produced from the segmentation software. A more detailed discussion of the investigation of these texture features may be found in [6].

5.2 Background for the Experiment

After the initial study of texture features described above, fractal methods were investigated for their potential to represent natural texture. As was demonstrated in the previous chapter, artificial fractal images of increasing dimension corresponded to increasing visual roughness. In addition to aerial images, two other categories of images

were studied: the synthetic fractal images generated in Chapter 4 and Brodatz textures. All images represented each pixel with one byte, allowing a maximum of 256 distinct gray shades.

Artificially generated fractal images provide the most objective standard for measuring fractal characteristics. Due to its natural appearance, two–dimensional fractional Brownian motion provides the basis of the synthetic fractal images. Using spectral synthesis, random midpoint displacement and random midpoint displacement with successive random additions, three sets of nine fractal images were generated, with dimensions 2.1 through 2.9. Images produced by each of the three synthetic algorithms were shown in Figures 32, 35 and 36, respectively. Five additional synthetic images were included in this category, shown in Figure 38: a constant gray shade, a diagonal linear gradient, uniformly distributed random "noise," Gaussian distributed random "noise" and the "homeplate" image. Used throughout the previous texture analysis described in section 5.1, the homeplate image consists of a homeplate–shaped region of Gaussian noise against a background or uniform noise. All of the artificial images were 256 rows by 256 columns, except for the "homeplate," which was 128×128.

Natural textures from the photographic album by Brodatz [3] have become an established standard for natural image texture analysis. These textures provide continuity between other texture research. Figure 39 illustrates some of the Brodatz textures used,
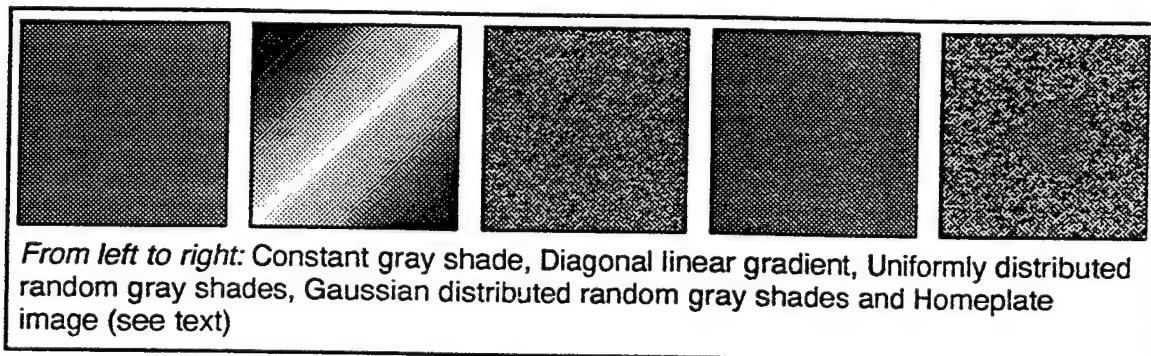

From left to right: Constant gray shade, Diagonal linear gradient, Uniformly distributed random gray shades, Gaussian distributed random gray shades and Homeplate image (see text)

Figure 38 Non–fractal Artificial Images

Top row, from left to right: Burlap, Clouds, Cork, Fieldstone, Grass
Bottom row, from left to right: Ice, Paper, Pigskin, Sand, Waves
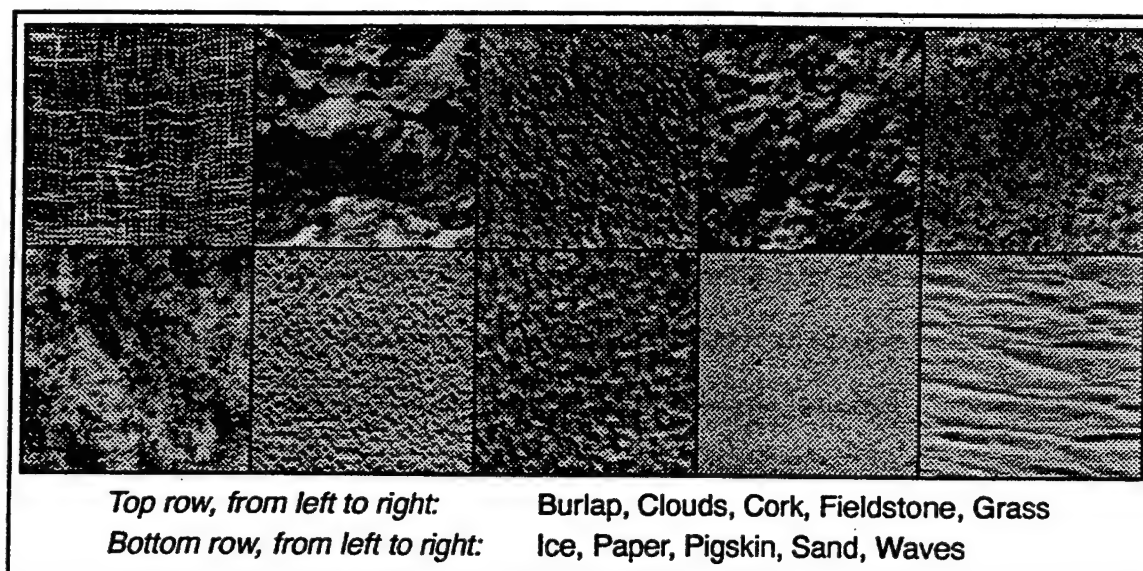
Figure 39 Sample Brodatz Textures

most of which have natural, fine–grained textures. The Brodatz textures were also
256×256 pixels in size, except for "clouds" and "waves," which were 241×256.

Finally, aerial imagery provides the focus and application of this research.
Although natural textures in aerial images have been studied, very little work can be
found which applies fractal analysis to this area. Scenes containing various mixtures of
natural features (such as water, trees, grass and dirt) and man–made features (such as
roads and buildings) were included in the analysis. The sizes of aerial images varied
widely. Most were larger than the 256×256 artificial and Brodatz textures.

Of the fractal characteristics mentioned in section 4.1, the most obvious is the
fractal dimension. Because a given texture may or may not be fractal in form, the degree
to which an image region fits the fractal model is a relevant consideration. Expressed in
terms of the root mean square (RMS) error between the predicted model and the actual
gray shade data, the "fractalness" of an image texture is explained further in section
5.5.3. The fractal error appears to have some potential use for distinguishing between
natural and man–made entities in an image, as natural objects are more apt to be fractal
than man–made objects. This application of the fractal error is examined in section 6.3.

A region descriptor, such as fractal dimension or fractal error, may be used either to characterize known regions or to segment an image into regions. Usually, the latter task is more difficult, since the regions must be identified as well as characterized. Accordingly, most research on fractal texture models focuses upon the analysis of known textured regions so that they may be distinguished from one another. However, the problem of segmentation using fractal characteristics has received less attention.

The characterization of a single texture may be considered as a "global" measure, in the sense that the measurement spans the entire image. In contrast, segmentation must describe the textural properties of smaller subregions of the image so that they may be distinguished from one another. This task may be considered a "local" measure, since it applies only to a particular neighborhood of the image. Both methods will be examined here: the description of a single texture (global measure) and the segmentation of an image containing several textures (local measure). Three global approaches will be discussed: spectral estimation, spatial estimation and box–counting. Local measurement will emphasize two spatial techniques. However, before attempting to estimate the fractal dimension either globally or locally, the relationship between the pixel spacing and the change in gray shade will be examined.

5.3 Fractal Scaling Property

Before estimating any fractal characteristics, the relationship between the pixel spacing and the change in gray shade across the image should be considered. This exponential relationship, sometimes called the fractal scaling property, was developed for fractional Brownian motion (fBm) in section 3.3.4:

$$E[\ |B_H(t_2) - B_H(t_1)| \ ] = k \ |t_2 - t_1|^H$$

Plotting the logarithm of the average change in the signal at a given interval versus the logarithm of that interval size will produce a straight line for fBm, the slope of which

will be the estimated value for the Hurst parameter H. (This slope corresponds to the spatial estimation method described in section 5.4.2.)

To reduce the computational effort, this comparison considered only the horizontal and vertical pixel distances in the image, for pixel distances one through twenty. The log–log plots for the synthetic fractal images were highly linear. These results are illustrated in Figure 40 for images with dimensions 2.1, 2.5 and 2.9 for each of the three fractal synthesis algorithms. The fit is especially close for the two spatial algorithms (random midpoint displacement with and without random additions); the difference between the plot and the estimated line is barely perceptible even if the graph is enlarged to fill the entire page. Log–log plots are also shown in Figure 40 for three of
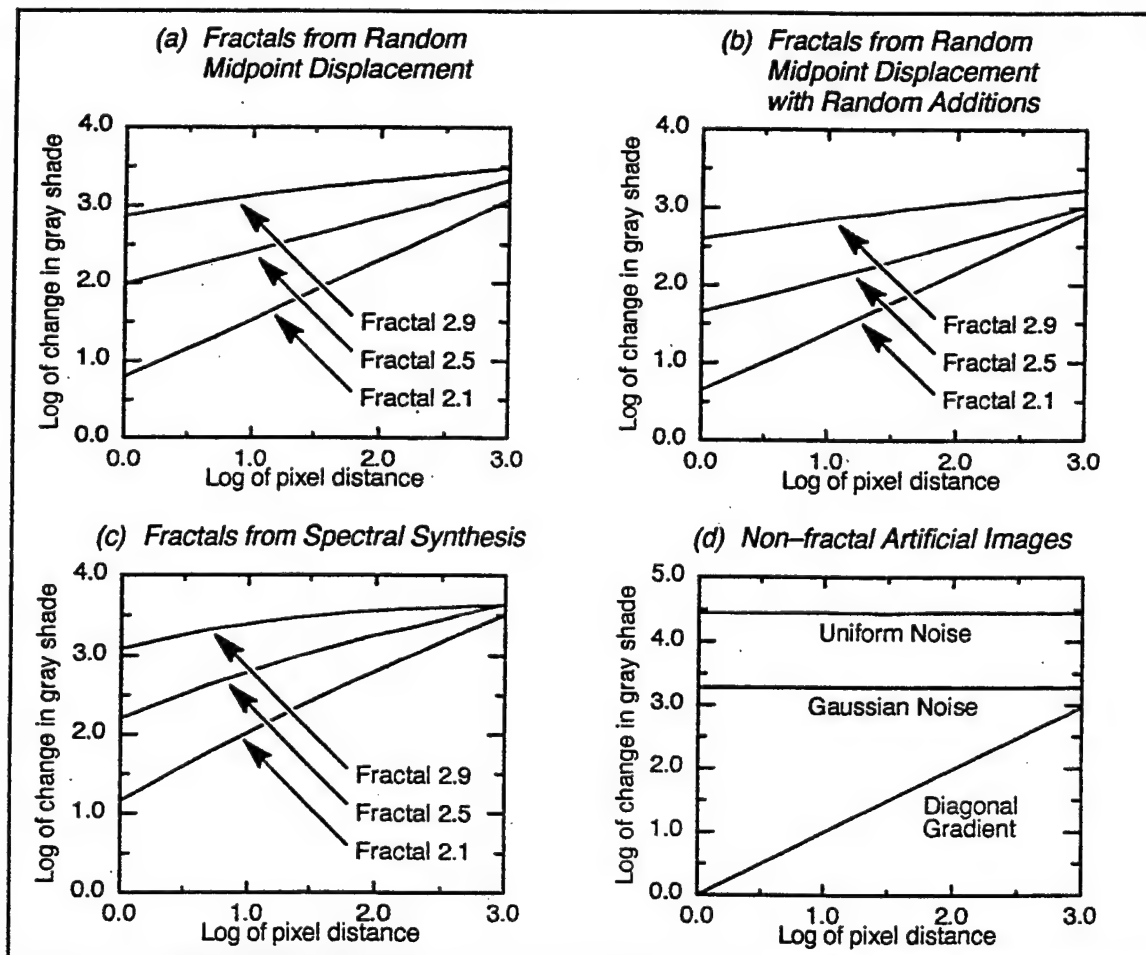

Figure 40 Pixel Distance Versus Change in Gray Shade for Artificial Images

the non–fractal artificial images, the diagonal gradient and the two random images. These also corresponded very closely to the fractal scaling property. The results for the Brodatz textures ranged from good for "clouds" to poor for "burlap." The graphs for the "clouds," "burlap," "cork" and "sand" textures are shown in Figure 41. The dashed lines indicate the regression lines of the data. Comparing the Brodatz textures visually to the artificial fractals, these results seem reasonable. The favorable relationship between the pixel spacing and the change in gray shade for the artificial fractals led to the next step, the global estimation of fractal characteristics.

5.4 Description of a Single Texture

5.4.1 Spectral Estimation of the Fractal Dimension

The spectral estimation of the fractal dimension stems from the relationship between the fractal dimension and the spectral distribution of the image, as described in section 4.3.1. The image may be converted to the frequency domain using a Fourier transform. Here, the power spectrum $S(f_1, f_2)$ of the frequency image of two–dimensional fractional Brownian motion is inversely proportional to the $H + 1$ power of the squared magnitude of the frequency:
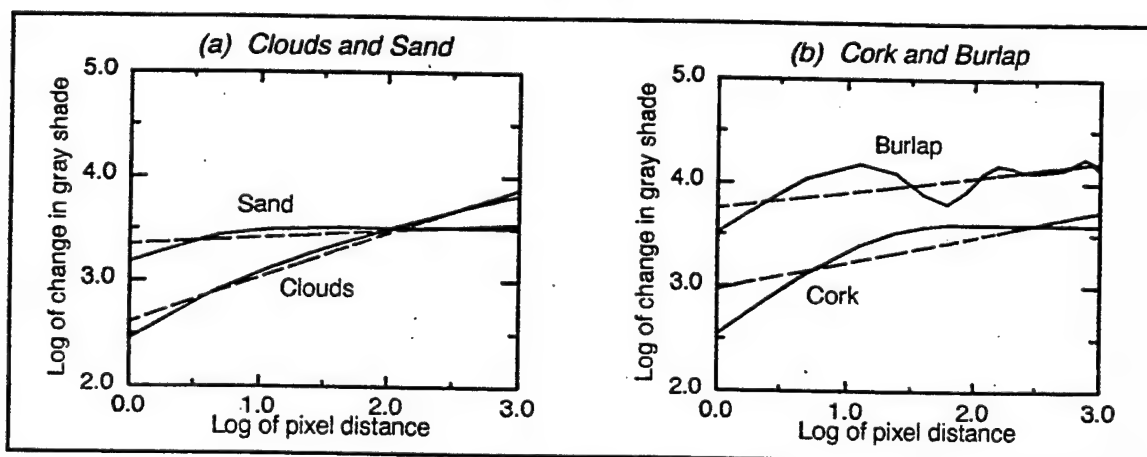

Figure 41 Pixel Distance Versus Change in Gray Shade for Brodatz Textures

$$S(f_1, f_2) = \frac{k}{(f_1{}^2 + f_2{}^2)^{H+1}}$$

In terms of the magnitude $V[n_1, n_2]$ of the discrete spectral image (which is the square root of the power spectrum), this relationship may be expressed as shown below, where n_1 and n_2 are the discrete frequency values:

$$V[n_1, n_2] = k\,(n_1{}^2 + n_2{}^2)^{-\frac{H+1}{2}}$$

Once the original image has been transformed to the frequency domain, the fractal dimension was estimated by fitting this spectral image to the above exponential equation. The relationship between the array indices and the frequency values n_1 and n_2 was shown in Figure 31. The Fourier coefficients $V[n_1, n_2]$ within this array can then be fit to the exponential equation above, and the parameters H (Hurst parameter) and k (proportionality constant) may be estimated. As mentioned previously in sections 4.3.1 and 3.4.5, the fractal dimension D can then be obtained readily from the Hurst parameter H using the equation $D = 3 - H$ (for two–dimensional signals).

Least squares linear regression was used to estimate parameters H and k from the frequency values n_1 and n_2 and the frequency magnitudes at these coordinates $V[n_1, n_2]$. The logarithm of the exponential equation gives a linear equation, shown below.

$$\log(V[n_1, n_2]) = \log(k) - \frac{H+1}{2}\,\log(n_1{}^2 + n_2{}^2)$$

A conventional least squares linear regression routine was implemented and tested thoroughly, including verification by sample manual calculations. This analysis was applied to the linear form of the exponential scaling formula, with the slope $-\dfrac{H+1}{2}$ yielding the Hurst parameter H. When estimating the dimension of a single texture, regression may be applied across the entire image to produce a single or global value for H and thus the fractal dimension. In addition to the fractal dimension, the root mean square (RMS) error in each estimate was also recorded.

For each of the three sets of artificial fractals, nine different dimensions were tested, from 2.1 to 2.9. For each of these dimensions, a set of ten images was generated,

using different random number seeds. As expected, the fractal dimension estimates were very close to one another within each set of ten statistically identical images, as were the RMS error values. The mean of the dimension estimates are graphed in Figure 42a for each of the three sets of artificial fractals, from 2.1 to 2.9. The dashed line indicates the ideal dimension estimate. The mean of the RMS error values are given in Figure 42b. Note that the RMS error indicates how well the data fits the prescribed model, not how accurate the dimension estimates were. Although the RMS error remained fairly constant for the spatial fractals, it actually decreased for the spectral fractals, in spite of the fact that the dimension estimates became slightly worse. With the exception of the diagonal gradient, the spectral estimates of all the other non–fractal images were nearly 4.0, clearly unreliable. The numeric data may be found in Tables 1–5 in section 5.4.4.

5.4.2 Spatial Estimation

Applying the scaling property of fBm to an image $G[\mathbf{x}]$, the average change in gray shade at a particular pixel distance is exponentially proportional to the magnitude of the pixel distance (see section 3.3.4):

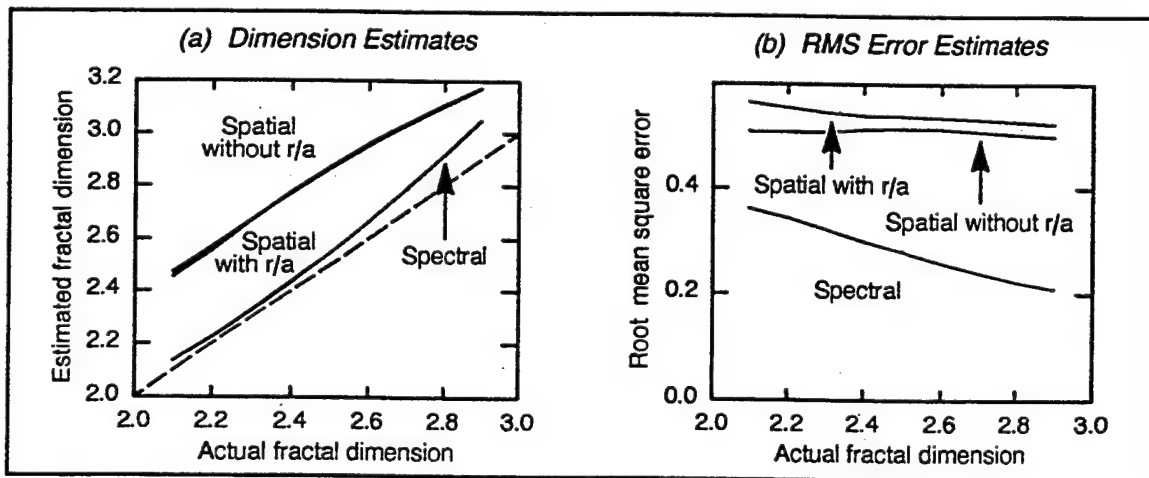$$E[\ |G[\mathbf{x}_2] - G[\mathbf{x}_1]|\ ] = k\ |\mathbf{x}_2 - \mathbf{x}_1|^H$$

Figure 42 Spectral Estimation of Synthetic Fractals

The fractal dimension of a two–dimensional signal will be $3 - H$. Following this equation, the change in gray shade may be averaged across several occurrences of a given distance. Collected for a set of pixel distances, this data may then be fit to the scaling property above using least squares linear regression on the logarithmic equation below:

$$\log(\ E[\ |G[x_2] - G[x_1]| \ ] \ ) = \log(k) + H \log(\ |x_2 - x_1| \ )$$

Examples of the logarithmic relationship between pixel distance and change in gray shade were shown in Figures 40 and 41. The closer the signal matches fBm, the more linear these graphs will be. For greater speed, only horizontal and vertical pixel spacings were used, as in section 5.3.

Ideally, the scaling property should apply across the entire range of pixel distances in the image. However, tests on the synthetic fractals indicated that the behavior became erratic for larger pixel distances, on average greater than fifty pixels. In fact, the change in gray shade actually dropped around a distance of one hundred pixels, after which it fluctuated gradually. Apparently, a limit exists on the range over which the signal may be considered uniformly fractal. Most observed fractal phenomena in nature also share such a bound, due to limitations inherent in the underlying physical processes. In addition to these difficulties, there will be fewer samples at larger pixel distances. Although there will be $N - 1$ times more samples at the minimum pixel distance of one than at the maximum distance of $N - 1$ for an $N \times N$ image (assuming only horizontal and vertical pixel spacings), each distance has equal weighting in the regression. Also, looking toward the goal of a local texture measure, the estimates at smaller pixel distances are of greater significance. Thus, the spatial estimates of fractal dimension and the root mean square (RMS) error were constrained to pixel distances one through twenty–five.

Figure 43a shows the estimated dimensions for the three sets of artificial fractals, each consisting of ten statistically identical images for each of nine dimensions from 2.1
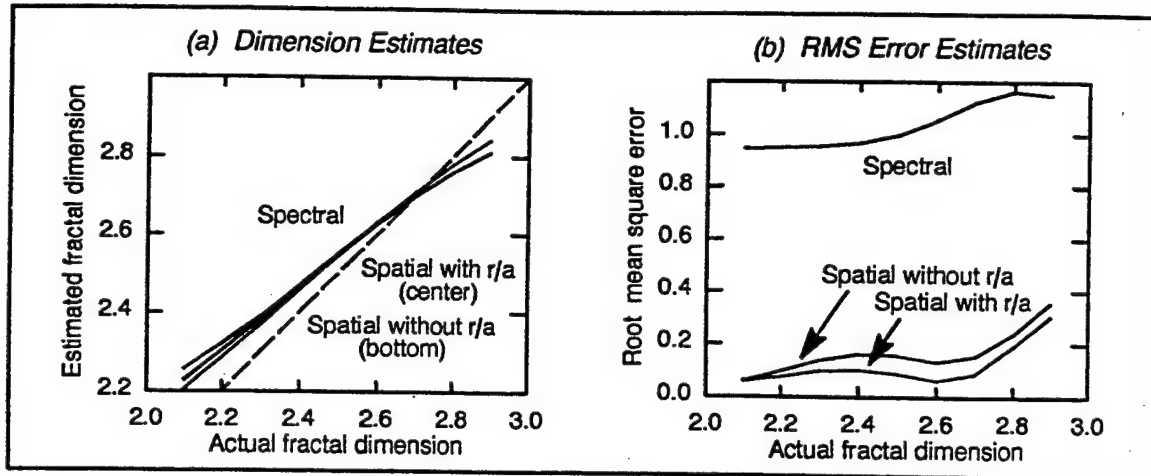
Figure 43 Spatial Estimation of Synthetic Fractals

to 2.9. (As before, the graphs include the averages for each set of ten images.) The ideal

dimension estimates would fall along the dashed line. The analysis of the random

midpoint displacement algorithm in section 4.4.1 indicated that theoretically valid results

should be obtained at the dimension 2.5. However, the estimation line crossed the ideal

line closer to an actual dimension of 2.7. The RMS error values for the same set of

images are graphed in Figure 43b. The RMS error remained somewhat constant for all

three sets of fractals, although it increased slightly as the actual dimension increased.

Recall that the RMS error indicates how well the data fits the prescribed model, not how

accurate the dimension estimates were. Also, these RMS error values cannot be

compared directly to those from the previous spectral estimation approach. For the

actual data and further discussion, see section 5.4.4.

The previous regression used the entire range of pixel values to obtain a single

estimate. The estimates could vary if a different range of pixel distances were used.

Accordingly, the fractal dimension and RMS error were calculated as functions of the

pixel distance. However, these measures cannot be computed for a single pixel distance,

since at least two data points are required for linear regression. Thus, two approaches

were tried, one using three pixel distances and the other using five. The estimate for the

pixel distance d relied upon distances $d - 1$, d and $d + 1$ in the former case, and

distances $d - 2$ through $d + 2$ in the latter case. Generally, the results from using three or five distances were similar, with slightly smoother functions obtained from the latter approach.

Ideally, the dimension should be constant across all pixel distances. This was true for the spatially synthesized fractals and the non–fractal artificial images. The dimension function increased slightly for the spectrally synthesized fractals, meaning that the estimated dimension would be higher in some cases over larger pixel spacings. These results are shown in Figures 44a and 44c for the spectral fractals and ordinary spatial fractals, respectively, using the five closest pixel distances for each estimate. (The images generated with random additions were excluded, since their results were very
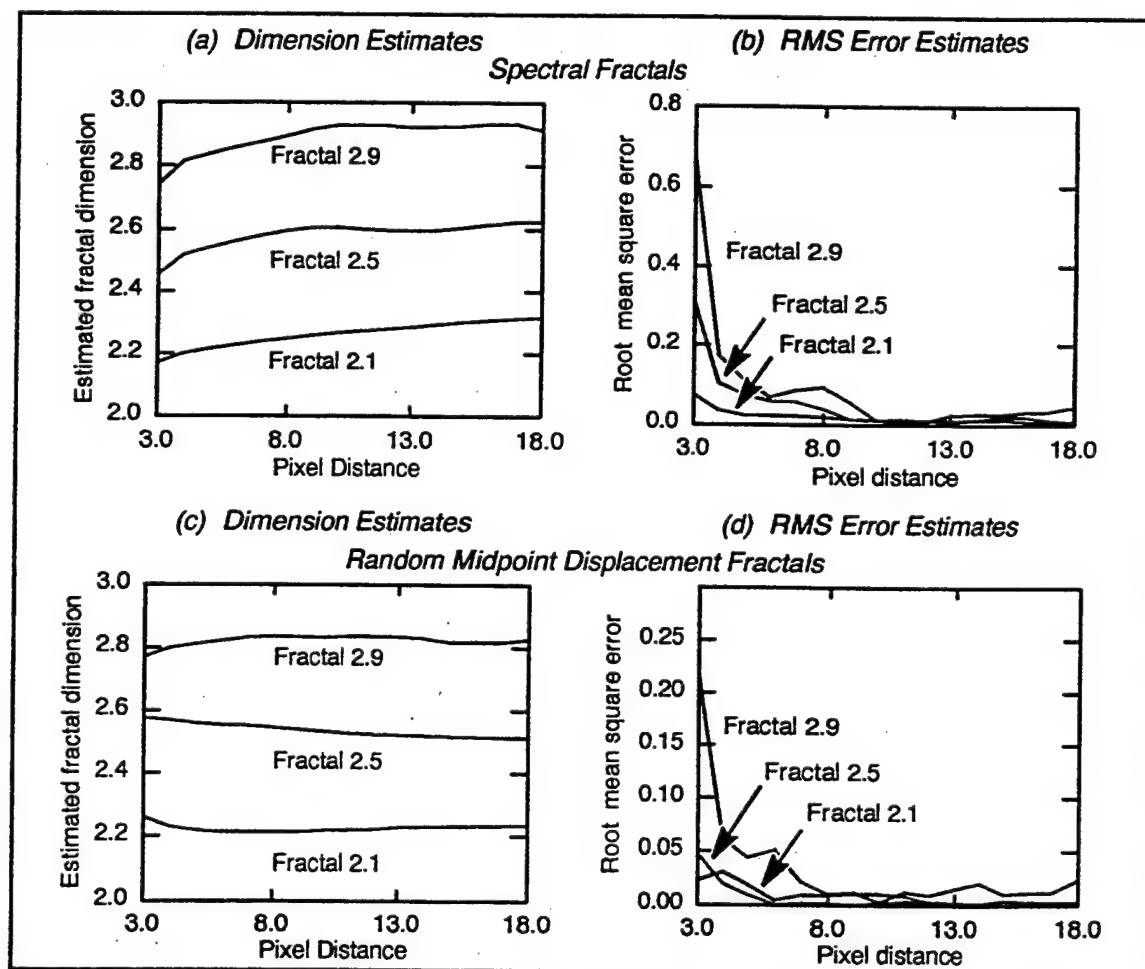

Figure 44 Dimension and RMS Error as Functions of the Pixel Distance

similar to those from the ordinary random midpoint displacement images.) The RMS error decreased quickly and remained the lowest for the ordinary spatial fractals. The RMS error was also low for the fractals with random additions. Here, the functions became somewhat stratified according to the dimension of the image. For example, the function for 2.9 remained completely above the function for 2.8. The spectral fractals produced the highest RMS error, with some stratification over lower pixel distances. The RMS error functions are graphed in Figures 44b and 44d for the spectral fractals and ordinary spatial fractals, respectively. As before, the five closest pixel distances were used.

Generally, the dimension curves were exceptionally flat for the non–fractal artificial textures, and the RMS error was typically very close to zero across the entire range. The notable exception was the uniform noise image. Although its RMS error began low, it rose above all the other fractal RMS error functions for larger pixel distances. The fractal dimension and thus the RMS error could not be computed for the constant image. The estimation takes the logarithm of the average change in gray shade, which will always be zero in this unusual case, leaving the logarithm undefined.

5.4.3 Box–Counting

A two–dimensional algorithm for estimating the box–counting dimension, described in sections 3.4.4 and 3.4.5, was implemented. The total number of boxes N_δ of a given size δ needed to cover a fractal follows an exponential relationship. The box–counting dimension D_B was defined in section 3.4.4 as

$$D_B = \lim_{\delta \to 0} \left\{ \log(N_\delta) / \log(1/\delta) \right\}$$

In practice, the box size cannot reach an arbitrarily small size for a discrete signal. Therefore, regression is performed on a set of small box sizes. Larger box sizes can adversely affect the estimate, since the number of large boxes needed to cover the object will level out. An arbitrary range of approximately the smallest ten percent of the

possible box sizes was used. For the synthetic fractals, having a size of 256 × 256 pixels, the maximum range of box sizes is from one to 256 pixels. Thus, box sizes one through twenty–five were used. Note that this corresponds to the range of pixel distances used with global spatial estimation in section 5.4.2. Lastly, although the original explanation of box counting was based upon integer values for N_δ, a fractional number of boxes was used here, for greater accuracy. Figures 45a and 45b show the dimension and root mean square (RMS) error, respectively, for each of the three sets of synthetic fractals. (Recall that the RMS error does not measure the accuracy of the dimension estimates. It indicates how well the data fits the model.) The fractal dimension estimates deviated from the ideal dimension (the dashed line in Figure 45a) as the actual dimension increased. At the same time, the RMS error values rose sharply, indicating that the data fit the model best for lower fractal dimensions. The RMS error values of the box–counting algorithm cannot be compared directly to those from either the spectral or spatial estimation algorithms, as will be explained in section 5.4.4.

Certain similarities are apparent between the box–counting and spatial estimation techniques. Both obtain an estimate from regression on an exponential equation. Although the actual exponents differ, both are directly related. (Recall that the box–counting model was used to determine the relationship between the fractal
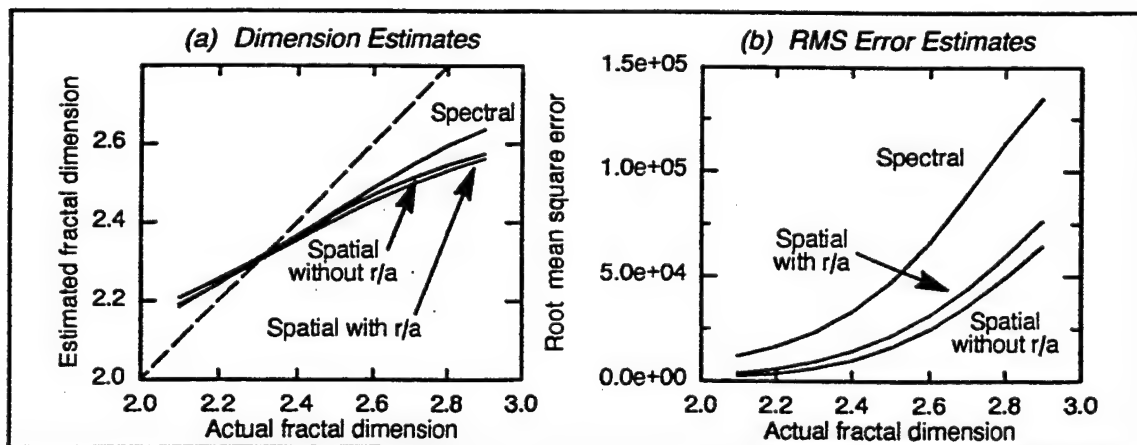

Figure 45 Box–Counting Estimation of Synthetic Fractals

dimension and the Hurst parameter, as used with spatial estimation.) The pixel spacings and box sizes correspond directly. Because a fractional number of boxes is allowed, it would appear at first that the number of boxes would match the average change in gray shade of the signal. Although this should be true in general, consider the following example, illustrated in Figure 46. Consider a pixel spacing and box size of four pixels. The spatial estimate will give a vertical displacement of one, ignoring the intermediate peak and valley. However, the box–counting method will not ignore the intermediate content. All points of the signal must be covered, even though the covering may be large and bulky. It is unclear, however, whether one approach is conceptually superior to the other. Instead, each approach is compared empirically in section 5.4.4.

5.4.4 Comparison of Results

Whereas each of the estimation algorithms was examined individually in the previous sections, here the three estimation algorithms will be compared against one another. Note that the RMS error values are not listed here. Although the RMS error values may be compared when using the same estimation algorithm, they may not be compared to those from another estimation algorithm. The RMS error is measured in terms of the power spectrum, the change in the signal and the number of boxes covering the signal for each the spectral, spatial and box–counting estimation algorithms, respectively. These quantities cannot be compared directly. This fact also illustrates why the magnitude of the RMS error differs for each estimation approach.
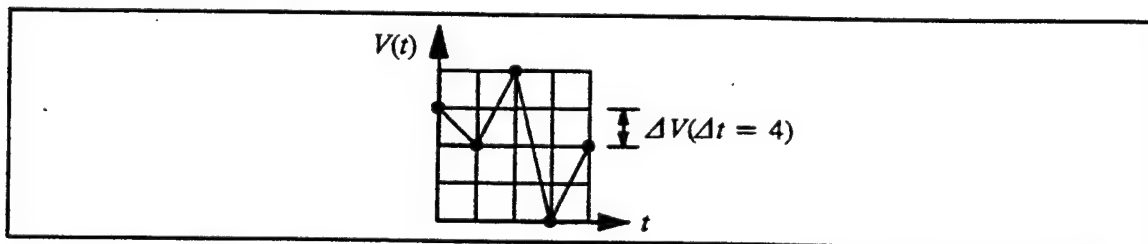

Figure 46 Conceptual Comparison of Box Counting and Spatial Estimation

In this study, ten images were generated with different random seeds for each of the nine dimensions 2.1 through 2.9, for a total of ninety images for each of the three synthesis techniques. The estimated fractal dimensions and RMS errors were highly consistent within each group of ten statistically identical images. Thus, only the averages within each group of ten images are considered in this discussion, since the individual values were always very close. On average, the variance of the fractal dimension values within each of the twenty–seven groups of ten images was only 0.00199 for spectral estimation, 0.00038 for spatial estimation and 0.00014 for box counting.

Table 1 lists the fractal dimension values of the spectrally synthesized fractals, with the absolute error between each estimated and actual value. The estimates from Keller et al. [25] are included also. (They offered the interpolated algorithm as an improvement over the standard box–counting technique.) The spectral values all overestimated the actual dimension, whereas the other methods began with overestimated values but ended with underestimated values. Judging by the average absolute error, the spectral and spatial algorithms performed well, comparable to the interpolated box–counting dimension presented in [25].

The midpoint displacement fractals, with and without random additions, are compared in Tables 2 and 3, respectively. As before, the fractal dimension and the absolute error between the estimated and actual dimensions are listed. Here, the spatial approach performs the best. Its estimates were too high for fractals with low dimension and too low for fractals with high dimension, but overall the absolute error was the least. The box–counting method had about twice the error as the spatial method. Its estimates began too high and ended much too low. Although the spectral approach gave excellent results in Table 1, it performed poorly with the spatial fractals. Its estimates were all much too large. Overall, the results between the spatial fractals with and without random additions were nearly identical for the spatial and box–counting estimation algorithms

Table 4 summarizes the results of each estimation algorithm, as applied to the three sets of synthetic fractals. The average absolute error between the estimated and actual dimensions indicates that the spatial estimation algorithm was consistently the best. Although the spectral approach worked well with spectrally synthesized fractals, it performed poorly on the spatial fractals. For the two sets of spatially synthesized fractals, the box–counting method did much better than the spectral approach. Although the RMS error values cannot be compared between different estimation algorithms, they may be compared within a single estimation approach. As expected, the fit was significantly better when the spectral approach was applied to spectral fractals and when the spatial approach was applied to spatial fractals, as opposed to when the two paradigms were mixed. Likewise, the RMS error values for the box–counting algorithm, which is essentially a spatial approach, were lower for the spatial fractals. Note that a good fit (low RMS error) can occur even if the actual estimates are in error (high error between the estimated and actual dimensions), as demonstrated by the box–counting results. Such a combination can be misleading. The RMS error indicates how well the data fits the model, not how close the dimension estimate might be.

The fractal dimension and RMS error values for the non–fractal artificial images are listed in Table 5. The constant and diagonal gradient images should have a fractal dimension of exactly two, which matched the spatial and box–counting estimates. (Technically, the spatial estimate for the constant image could not be computed, as explained in section 5.4.2.) Although the spectral estimates of the diagonal gradient image were within an acceptable range, the value for the constant image was clearly incorrect. The noise images should, in some sense, fill the three–dimensional space. The uniformly distributed random signal should fill this space more densely than the Gaussian signal, which was reflected by the box–counting estimates. The spatial estimates were both near 3.0, but the spectral estimates were much too high. Generally, the RMS error values of the non–fractal artificial images were greater than those for the

fractal images. The exception was spatial estimation, whose values were comparable for both the fractal and non–fractal artificial images.

Table 6 lists the fractal dimension and RMS error values for the Brodatz textures pictured in Figure 39. Consistently, the fit was the best for the "clouds," "sand" and "waves" textures and worst for "burlap" and "paper." The dimension estimates for the spatial estimation and box–counting algorithms were consistent with those produced for the fractal images. However, the spectral dimension estimates did not correspond very well to the values from the other two estimation approaches. For example, consider a synthetic fractal with dimension 2.6 and the "clouds" texture. The three synthetic fractals with dimension 2.6 produced spatial estimates ranging between 2.62 and 2.64 and box–counting estimates ranging between 2.45 and 2.49. These two estimation ranges are very close to the estimates obtained for the "clouds" texture, 2.60 and 2.48, respectively. In contrast, the spectral estimates for synthetic fractals with an actual dimension of 2.6 ranged from 2.66 to 2.98, which is not even close to the spectral estimate of 2.17 for the "clouds" texture. In all cases, the RMS error values of the Brodatz textures were significantly higher than those obtained from the synthetic fractals, as would be expected. This particular group of fine–grained Brodatz textures produced RMS error values which were six times greater than those of the fractal images, on average. This would seem to indicate that, in spite of the variations in the RMS error values for the synthetic fractals, the fractal textures all produced a comparatively close fit.

Of the three global estimation algorithms implemented, the spatial approach gave the best results. Its dimension estimates were closest to the actual dimensions of all three sets of artificially synthesized fractal textures. Its average absolute error was slightly lower than the improved interpolated box–counting algorithm of Keller et al. [25], although the individual absolute errors (see Table 1) of the spatial method were less uniform than those of the interpolated approach. The dimension estimates from the

non–fractal artificial textures and the Brodatz textures appeared to correspond well with their apparent visual roughness. Therefore, the spatial estimation approach was chosen as the predominant paradigm for further development into a local texture measure.

Table 1 Global Fractal Dimension Estimates of Spectrally Synthesized Fractals

Orig. Dim.	Spectral		Spatial		Box Counting		Standard Box Counting [25]		Interpolated Box Ct. [25]	
	F dim	|error|	F dim	|error|	F dim	|error|	F dim	|error|	F dim	|error|
2.1	2.136	0.036	2.256	0.156	2.208	0.108	2.12	0.02	2.16	0.06
2.2	2.224	0.024	2.319	0.119	2.256	0.056	2.18	0.02	2.26	0.06
2.3	2.326	0.026	2.390	0.090	2.309	0.009	2.25	0.05	2.38	0.08
2.4	2.435	0.035	2.468	0.068	2.367	0.033	2.34	0.06	2.49	0.09
2.5	2.547	0.047	2.549	0.049	2.426	0.074	2.42	0.08	2.59	0.09
2.6	2.662	0.062	2.631	0.031	2.486	0.114	2.48	0.12	2.67	0.07
2.7	2.786	0.086	2.709	0.009	2.543	0.157	2.52	0.18	2.73	0.03
2.8	2.915	0.115	2.780	0.020	2.594	0.206	2.53	0.27	2.77	0.03
2.9	3.049	0.149	2.843	0.057	2.636	0.264	2.53	0.37	2.81	0.09
Aver. |error|		0.064		0.066		0.114		0.130		0.067

Table 2 Global Fractal Dimension Estimates of Spatially Synthesized Fractals, Random Midpoint Displacement

Orig Dim	Spectral		Spatial		Box Counting	
	Fractal Dim	|error|	Fractal Dim	|error|	Fractal Dim	|error|
2.1	2.453	0.353	2.206	0.106	2.184	0.084
2.2	2.554	0.354	2.284	0.084	2.243	0.043
2.3	2.667	0.367	2.369	0.069	2.306	0.006
2.4	2.775	0.375	2.458	0.058	2.366	0.034
2.5	2.871	0.371	2.545	0.045	2.421	0.079
2.6	2.959	0.359	2.626	0.026	2.470	0.130
2.7	3.037	0.337	2.698	0.002	2.511	0.189
2.8	3.108	0.308	2.760	0.040	2.545	0.255
2.9	3.174	0.274	2.812	0.088	2.573	0.327
Average |error|		0.344		0.058		0.128

Table 3 Global Fractal Dimension Estimates of Spatially Synthesized Fractals, Random Midpoint Displacement with Random Additions

Orig Dim	Spectral		Spatial		Box Counting	
	Fractal Dim	|error|	Fractal Dim	|error|	Fractal Dim	|error|
2.1	2.472	0.372	2.228	0.128	2.190	0.090
2.2	2.569	0.369	2.302	0.102	2.244	0.044
2.3	2.670	0.370	2.383	0.083	2.301	0.001
2.4	2.770	0.370	2.467	0.067	2.355	0.045
2.5	2.864	0.364	2.550	0.050	2.407	0.093
2.6	2.951	0.351	2.628	0.028	2.454	0.146
2.7	3.030	0.330	2.698	0.002	2.495	0.205
2.8	3.104	0.304	2.760	0.040	2.530	0.270
2.9	3.172	0.272	2.811	0.089	2.561	0.339
Average |error|	0.345		0.065		0.137	

Table 4 Summary of Global Estimation of Fractal Images

Type of Fractal	Spectral		Spatial		Box Counting	
	aver |err|	avg RMS	aver |err|	avg RMS	aver |err|	avg RMS
Spectral fractals	0.0645	0.2826	0.0665	1.0350	0.1135	59639
Spatial without r/a	0.3443	0.5087	0.0575	0.1656	0.1275	23671
Spatial with r/a	0.3450	0.5403	0.0654	0.1190	0.1370	29635

Table 5 Global Estimates of Non–Fractal Artificial Images

Image	Spectral		Spatial		Box Counting	
	F Dim	RMS err	F Dim	RMS err	F Dim	RMS err
Constant	3.999	1.118	—	—	2.024	0.022
Diagonal Gradient	2.148	0.685	2.016	0.082	2.040	810
Uniform Noise	3.991	0.517	3.000	0.162	2.915	511678
Gaussian Noise	3.987	0.500	3.000	0.038	2.753	128031
Homeplate	3.958	1.034	3.000	0.332	2.845	96508

Table 6 Global Estimates of Brodatz Textures

Image	Spectral		Spatial		Box Counting	
	F Dim	RMS err	F Dim	RMS err	F Dim	RMS err
Burlap	2.075	2.133	2.852	5.824	2.754	481307
Clouds	2.165	1.073	2.595	1.400	2.475	78809
Cork	2.089	1.079	2.789	3.875	2.527	143003
Fieldstone	2.121	1.153	2.636	3.780	2.455	117131
Grass	2.048	1.561	2.782	3.474	2.531	141653
Ice	2.133	1.717	2.753	2.533	2.583	189673
Paper	2.001	2.700	2.856	5.709	2.623	274217
Pigskin	2.096	1.533	2.776	3.977	2.533	153846
Sand	3.233	0.701	2.946	1.395	2.617	159790
Waves	2.431	0.699	2.686	2.278	2.503	86356

5.5 Segmenting an Image

5.5.1 Developing a Local Measure

In addition to the qualifications needed for a global measure, a local texture measure must characterize a small or local region of the image. This characterization must be capable of distinguishing between desired categories of local regions in order to segment or subdivide the image. The local texture measure must consider an area large enough to include a reasonable sampling of a texture, but small enough to isolate one particular texture among whatever other textures might be present in the image. Neighborhoods whose sizes ranged from 5×5 through 17×17 pixels were studied, but the smaller sizes appeared more suitable for the natural textures in aerial images.

Based upon the results of global estimation, the spatial estimation algorithm was chosen as the primary focus of fractal-based local texture measures. However, a local spectral method was tested first, following the approach used in Pentland's influential work [39]. His selection of an 8×8 square of pixels, or window, was appropriate for this application. Also, since this size is a power of two, the fast Fourier transform may be used. To reduce computational demands, non-overlapping windows (see section

2.3.3) were used. Each non–overlapping 8 × 8 window of the image was individually transformed to the frequency domain. The resulting 8 × 8 spectral image was analyzed by the least squares linear regression algorithm presented in section 5.4.1 to produce an estimate of the fractal dimension for that portion of the image.

This approach was applied to the artificial textures described in section 5.2. For an image with 256 × 256 pixels, there were 32 × 32 windows, each with 8 × 8 pixels. The fractal dimension estimates for the 1024 windows were collected together. The mean and standard deviation of the fractal dimension estimates are listed in Table 7 for each of the three sets of synthetic fractals, along with the absolute error between the estimated and actual dimensions.

Table 7 Local Spectral Estimates of Fractal Images

Orig Dim	Spectral			Spatial, without r/a			Spatial, with r/a		
	Dim μ	Dim σ	|error|	Dim μ	Dim σ	|error|	Dim μ	Dim σ	|error|
2.1	1.847	0.314	0.253	2.168	0.269	0.068	2.179	0.262	0.079
2.3	2.077	0.302	0.223	2.375	0.251	0.075	2.359	0.247	0.059
2.5	2.344	0.316	0.156	2.582	0.260	0.082	2.551	0.237	0.051
2.7	2.624	0.294	0.076	2.761	0.258	0.061	2.728	0.242	0.028
2.9	2.926	0.305	0.026	2.917	0.259	0.017	2.890	0.238	0.010
Average |error|		0.147			0.061			0.045

The accuracy was surprisingly good, especially considering that these measurements came from 8 × 8 windows, in contrast to the entire image for global estimation. Comparing the spectrally synthesized fractals, the average absolute error of the local spectral estimates (Table 7) was only about twice that of the global spectral estimates (Table 4). For the spatially synthesized fractals, the local dimension estimates were actually better than the global dimension estimates, by a considerable amount. This could indicate that the lower frequency information used by the global algorithm actually harmed the spectral estimates, similar to the manner in which the spatial and

box–counting estimation algorithms suffered when the pixel distances became too large. However, the local spectral estimates were poor for the non–fractal artificial images. The actual dimension of 2.0 for the constant and diagonal gradient images was overestimated at 3.6 and 3.5, respectively. Likewise, the estimates for the uniform and Gaussian noise images were much too high.

As applied to segmentation, the fractal dimension of the local spectral approach did not provide enough distinction. The best result was obtained for a collage of three spectrally synthesized fractals, shown in Figure 47a. The top half has a dimension of 2.1, the bottom half has a dimension of 2.9 and the center has a dimension of 2.5. The fractal dimension image was enlarged and normalized from the range [0.95, 4.03] to [0, 255] so that it could be displayed as a gray shade image in Figure 47b. Dark pixels indicate a low fractal dimension; light pixels represent a high dimension. The average estimated dimensions for the top (2.1), bottom (2.9) and center (2.5) regions were 1.92, 2.95 and 2.46. For aerial images, the local spectral measure of fractal dimension could only detect the faint impressions of very large features. For example, the fractal dimension image of

(a) Original Image   (b) Fractal Dimension Image
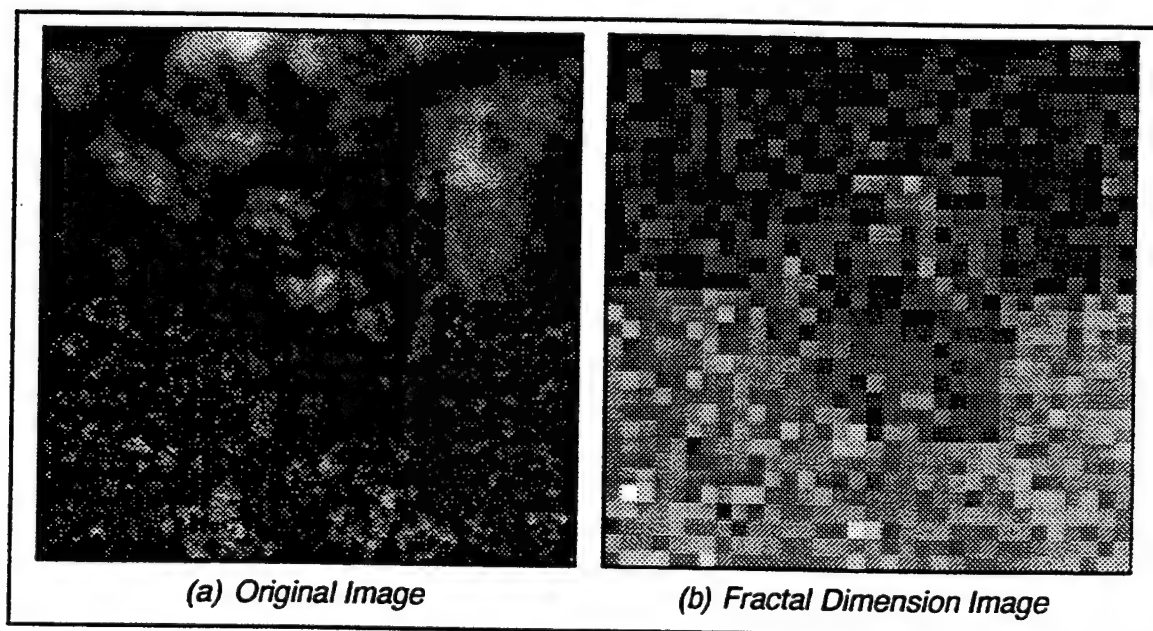
Figure 47 Spectral Fractal Dimension of Fractal Collage Image

the shore scene (Figure 48a), enlarged and normalized from the range [0.27, 3.32], is shown in Figure 48b. At best, the land and water regions are vaguely discernable. (Also, a slight difference is noticeable at the bottom and right edges, where there were not enough pixels to make a complete 8×8 square.)

Although the local spectral estimation method gave exceptionally good results for the synthetic fractal images, its performance suffered on the non–fractal artificial images. Its ability to segment images was low. Certainly, this approach warrants further investigation. However, a faster algorithm was desired, that would allow overlapping windows for greater resolution.

5.5.2 Spatial Estimation

The computationally complexity of the Fourier transform makes spectral techniques less attractive. For a window size that is a power of two, the fast Fourier transform (FFT) may be used, which significantly increases the speed. For a window

(a) Original Image   (b) Fractal Dimension Image
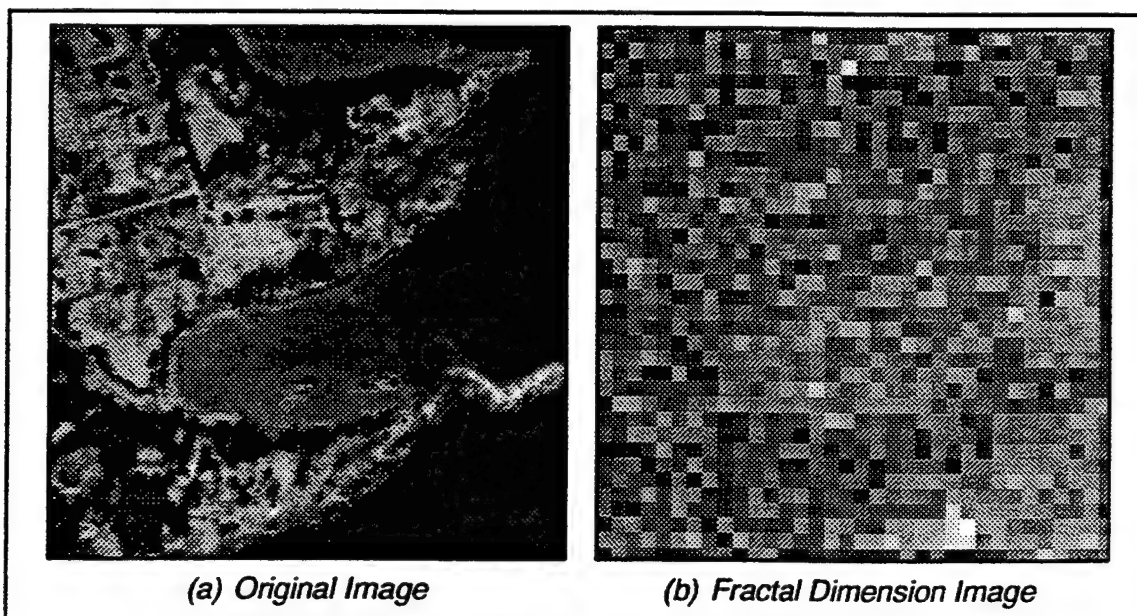
Figure 48 Spectral Fractal Dimension of Shore Scene

with $w \times w$ pixels, a spectral method will require $w^2 \log^2(w)$ computations if w is a power of two and w^4 computations otherwise.

Spatial measures offer an alternative. However, there is a total of $(w^4 - w^2)/2$ pixel distances within a window of size $w \times w$. Although this improves upon the ordinary Fourier transform, it is much worse than the FFT. Two improvements were examined. By considering only those pixel spacings in the horizontal and vertical directions, only $w^3 - w^2$ data points must be used. This approach was used with global spatial estimation in section 5.4.2. A further reduction results from a second technique. It includes all distances and angles within the window, but only with respect to the center pixel. There are $w^2 - 1$ such pixel spacings. Figure 49 compares the algorithmic complexities of the four estimation approaches described here. The spectral curve dips at pixel distances which are powers of two, where the FFT may be used.

The horizontal and vertical pixel spacings that would be used for a 4×4 window are illustrated in Figure 50a. (The center–oriented spacings in Figure 50b will be discussed later.) In this example, there are twenty–four spacings with a distance of one pixel, sixteen with a distance of two pixels and eight with a distance of three pixels. Although windows between 5×5 and 25×25 pixels were considered, smaller, odd–numbered sizes received the most attention. The estimated dimensions for each of
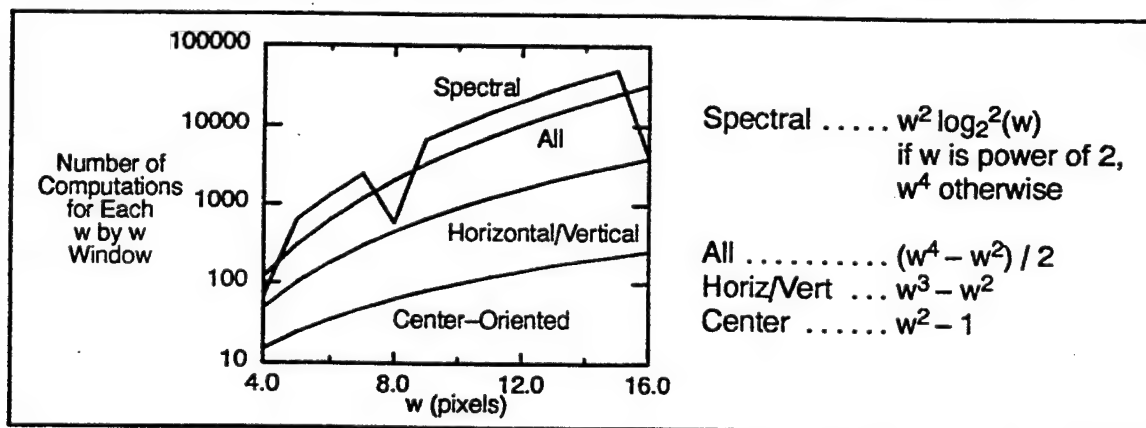

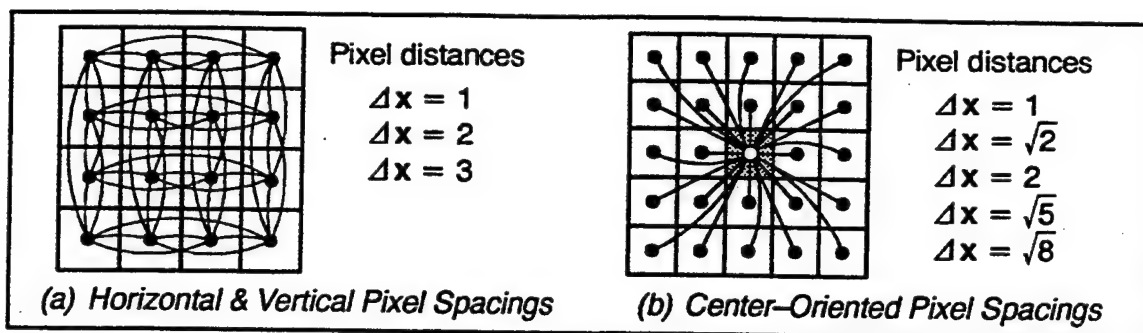Figure 49 Algorithmic Complexities for Various Estimation Approaches

Figure 50 Pixel Spacings Used with Local Spatial Estimation

the three sets of synthetic fractals are graphed in Figure 51. Data from window sizes of
5×5, 7×7 and 9×9 are shown in Figures 51a, 51b and 51c, respectively. The
dashed lines indicate the ideal dimension estimates. In all cases, the accuracy increased
with the window size, in terms of the average absolute error between the estimated and
actual dimensions. However, the RMS error also increased for larger window sizes.
Although this finding appears to contradict expectations, it is consistent with the global
spatial estimates. Over 90% of the synthetic fractals experienced an increase in RMS
error between estimates which used pixel distances one through five and those which
used distances one through ten. (Recall that the RMS error in Figures 44b and 44d
tended to decrease sharply for distances under about four or six pixels, after which it
became nearly level.) Table 8 lists the average absolute error between the estimated and
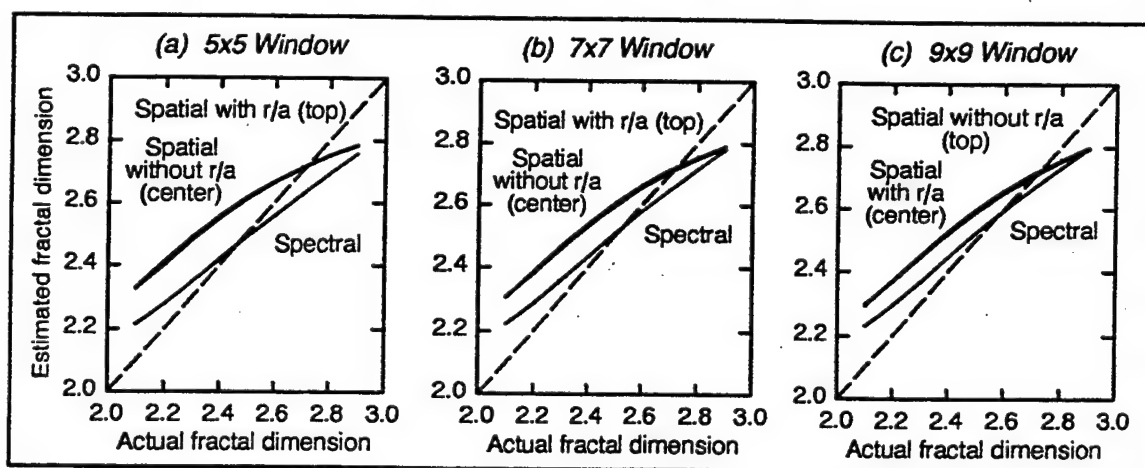
Figure 51 Local Dimension Estimates from Horizontal/Vertical Approach

actual dimensions and the average RMS error for each of the three sets of synthetic fractals and each of the three window sizes. The accuracy compared well against the global estimation algorithms, producing comparable average RMS error values.

Table 8 Summary of Local Estimation of Fractal Images,
from Horizontal/Vertical Approach

Type of fractal	5x5 window		7x7 window		9x9 window	
	aver |err|	avg RMS	aver |err|	avg RMS	aver |err|	avg RMS
Spectral	0.071	1.028	0.065	1.284	0.063	1.448
Spatial without r/a	0.127	0.758	0.115	0.860	0.107	0.927
Spatial with r/a	0.123	0.570	0.111	0.651	0.104	0.696

The horizontal and vertical method was next applied to actual images. The dimension images were normalized to the range [0, 255] so that they could be displayed as gray shade images. (Note that direct visual comparisons between different dimension images will not be meaningful, since each dimension image was individually normalized.) The results were not much better than those obtained with local spectral estimation. For example, Figure 52a shows the dimension image of the fractal collage (Figure 47a), using a 7×7 window. (The image was normalized from the range [1.98, 3.34].) The top and bottom regions were separated, but the middle region is not especially distinct. Many linear artifacts are apparent, presumably related to the horizontal and vertical orientation of the measurements. Arguably, the window size could be too small to identify the fractal textures adequately. Having much smaller features, the shore scene of Figure 48a was examined. Normalized from the range [1.98, 4.07], the dimension image is shown in Figure 52b, generated using a 5×5 window. Although there is some distinction between the land and water regions, the separation is unsatisfactory. Similar results were obtained for other aerial images.

(a) Fractal Collage (7x7 Window) *(b) Shore Scene (5x5 Window)*
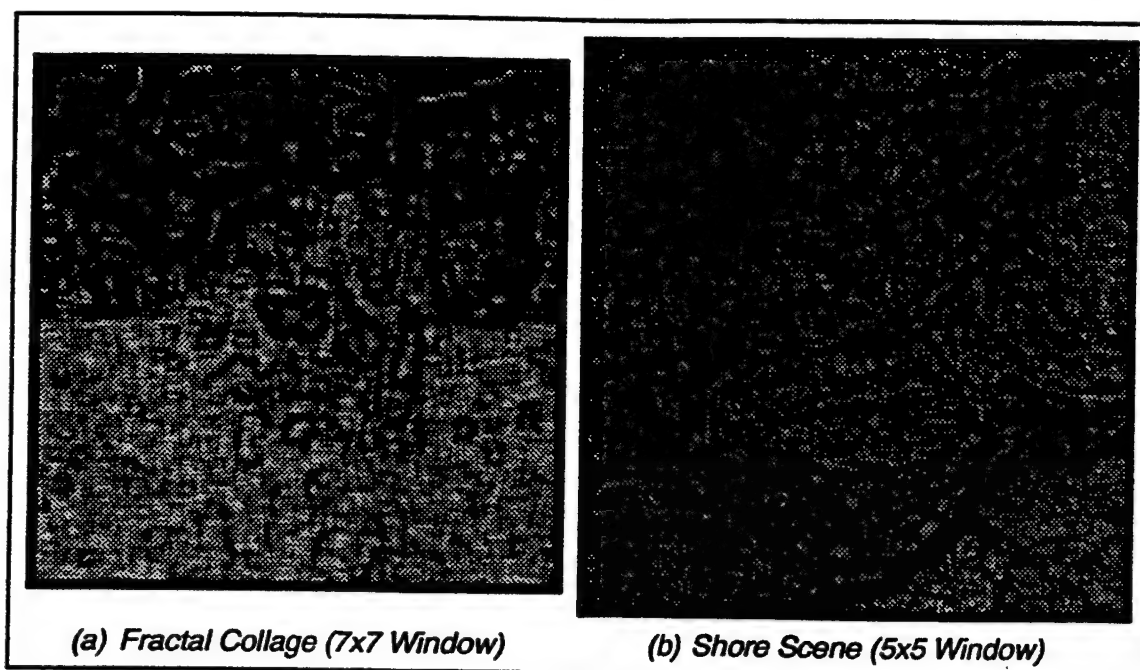
Figure 52 Dimension Images from the Horizontal/Vertical Approach

A better and faster alternative to the horizontal and vertical pixel spacing algorithm was sought. This led to the center–oriented pixel spacing method, illustrated in Figure 50b for a 5×5 window. In this example, there are four spacings each for pixel distances 1, $\sqrt{2}$, 2 and $\sqrt{8}$, plus eight spacings with a distance of $\sqrt{5}$. Thus, for a 5×5 window, there are a total of twenty–four center–oriented pixel distances, compared with one hundred horizontal and vertical pixel distances for the same size window. Intuitively, the center–oriented approach seems to localize the measure more by emphasizing the center pixel, plus it strives to make the measure more directionally independent than the former approach.

Most testing used 5×5, 7×7 and 9×9 window sizes. Figure 53a shows the average estimated fractal dimension for each of the three sets of synthetic fractals, using the 5×5 center–oriented algorithm. Figures 53b and 53c show the graphs for 7×7 and 9×9 window sizes. The ideal estimates would fall along the dashed line. Similar to the results from the horizontal and vertical method, the average absolute error between the estimated and actual fractal dimensions decreased as the window size increased. At
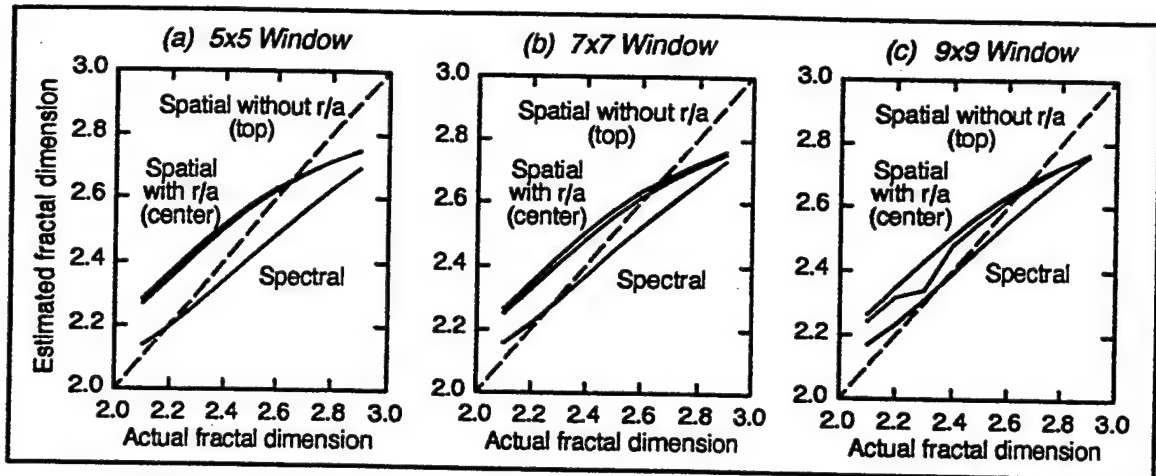
Figure 53 Local Dimension Estimates from Center–Oriented Approach

the same time, the average RMS error generally increased. Table 9 lists the average

absolute error and average RMS error for each of the three sets of synthetic fractals,

tested by each of three different window sizes. The accuracy was again comparable to

the global estimation algorithms. Except for the spectrally synthesized fractals, the

center–oriented approach produced smaller average absolute errors than the horizontal

and vertical approach. However, the RMS error values were significantly higher in all

cases for the center–oriented technique, probably because fewer pixel spacings

contributed to the estimation.

Table 9 Summary of Local Estimation of Fractal Images,
from Center–Oriented Approach

Type of fractal	5x5 window		7x7 window		9x9 window	
	aver |err|	avg RMS	aver |err|	avg RMS	aver |err|	avg RMS
Spectral	0.095	2.418	0.072	7.701	0.059	3.800
Spatial without r/a	0.109	1.945	0.099	2.469	0.092	2.779
Spatial with r/a	0.100	1.537	0.089	2.007	0.076	2.157

Applied to actual images, the center–oriented method was not a significant

improvement. The dimension image of the fractal collage (Figure 47a) was normalized

from the range [0.24, 4.41] and is shown in Figure 54a. A 7 × 7 window was used.
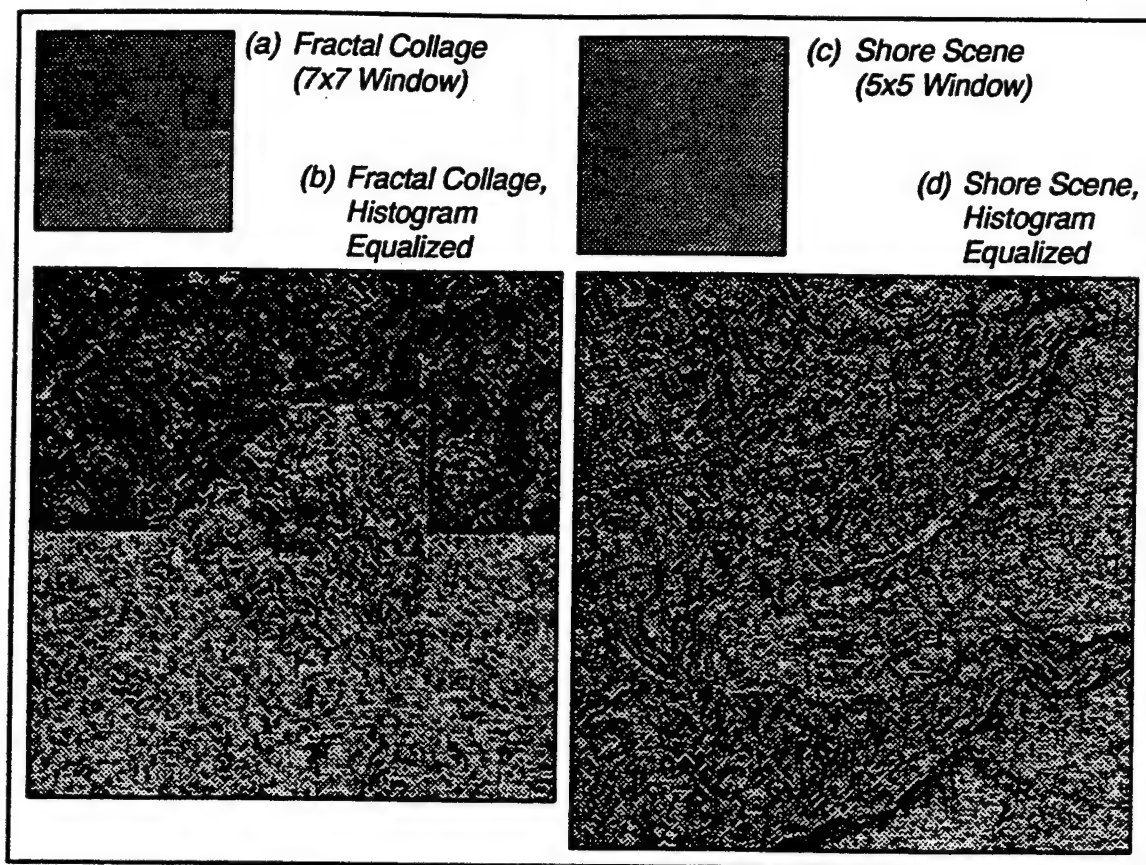
Figure 54 Dimension Images from the Center–Oriented Approach

The same image is shown in Figure 54b after histogram equalization, to facilitate visual analysis. Although the segmentation is better than that from the horizontal and vertical method (Figure 52a), the improvement is not substantial. The dimension image of the shore scene (Figure 48a) used a 5×5 window. Normalized from the range [−1.47, 5.13], the result is shown in Figure 54c. Figure 54d shows the dimension image after histogram equalization. However, the quality remained unsatisfactory. The dimension operator appeared to extract second–derivative information from the original image, which was not helpful for this task. Nevertheless, the center–oriented approach produced equivalent or slightly improved results than the horizontal and vertical method, plus it required less computation.

Note that both local spatial estimation approaches assume that all windows contain a fractal texture. Even if this were the case, boundaries between two regions

with different fractal dimensions would likely be non–fractal. Thus, problems will inevitably occur when a fractal estimate is taken within a window which does not have a uniform fractal dimension. This problem is addressed further in the following section.

5.5.3 Spatial Estimation of the Fractal Error

The regions observed in an actual image may be a combination of a wide variety of textures, some fractal and some not fractal. Before assuming that a region is fractal and attempting to extract its dimension, a relevant prerequisite is to first determine whether or not the region may be considered a fractal. One such approach is to measure the error produced when estimating the fractal dimension. A small error indicates that the observed data fits the fractal model well and thus may be considered fractal, whereas a large error means that the examined region is not fractal and thus the fractal dimension estimate is unreliable. This insight led to the inclusion of fractal error measurements throughout this study, for both global and local estimation.

Due to their greater processing speed, the two previous spatial models were chosen over the spectral model for estimating the fractal dimension. As before, the fractal dimension was estimated within each overlapping window of the image. The first approach used all possible horizontal and vertical distances within the window. The second approach used the set of gray shade differences which all include the center pixel of the window. The error between the estimated fractal dimension of a window and its actual data was used to quantify the degree to which that window may be considered a fractal texture. As explained in section 5.4.2, the average absolute change in gray shade of fractional Brownian motion at a given distance is proportional to the H power of the magnitude of that distance:

$$E[\ |G[\mathbf{x}_2] - G[\mathbf{x}_1]|\ ] = k\ |\mathbf{x}_2 - \mathbf{x}_1|^H$$
$$E[\ |\Delta G_{|\Delta\mathbf{x}|}|\ ] = k\ |\Delta\mathbf{x}|^H$$

The estimated values $\overline{H}$ and $\overline{k}$ of the parameters H and k may be obtained using least squares linear regression on the logarithm of the above equation. The estimated average absolute change in gray shade, at a given pixel distance, may be computed from the estimates $\overline{H}$ and $\overline{k}$. The difference between the actual value of the average absolute change in gray shade and this estimate is the error.

$$error_{|\Delta x|} = E[\,|\Delta G_{|\Delta x|}|\,] - \overline{k}\,|\Delta x|^{\overline{H}}$$

With the horizontal and vertical method, there will be $w - 1$ pixel distances and thus $w - 1$ error values. With the center–oriented approach, there will be five, nine or fourteen error values, for windows of size five, seven or nine. A variety of error measurements are possible, but the root mean square (RMS) error tends to be a more rigorous and accepted form than others (e.g., average absolute error). The RMS error is obtained from the square root of the average of the squared error values, where n is the number of error values, as shown below:

$$RMS\ error = \sqrt{\frac{1}{n}\sum_{|\Delta x|}(error_{|\Delta x|})^2}$$

Such a value is calculated for each overlapping window. Statistics may be gathered on the array of data or the result may be normalized to the range [0, 255] and displayed as a gray shade image.

Interestingly, the RMS error is somewhat distinct for synthetic fractals of different dimensions for small pixel distances, before the RMS error settles to a stable range. Thus, the fractal error could separate the three regions within the fractal collage (Figure 55a). Figure 55b shows the fractal error image of the fractal collage, using the horizontal and vertical method. The corresponding image for the center–oriented method is shown in Figure 55c. A 7×7 window size was used in both cases, and the error images were normalized from the ranges [0.02, 11.36] and [0.18, 34.46], respectively. Note that the fractal error images are negated to improve their visual clarity. Thus, light areas indicate low error or high "fractalness," whereas high error or low "fractalness"
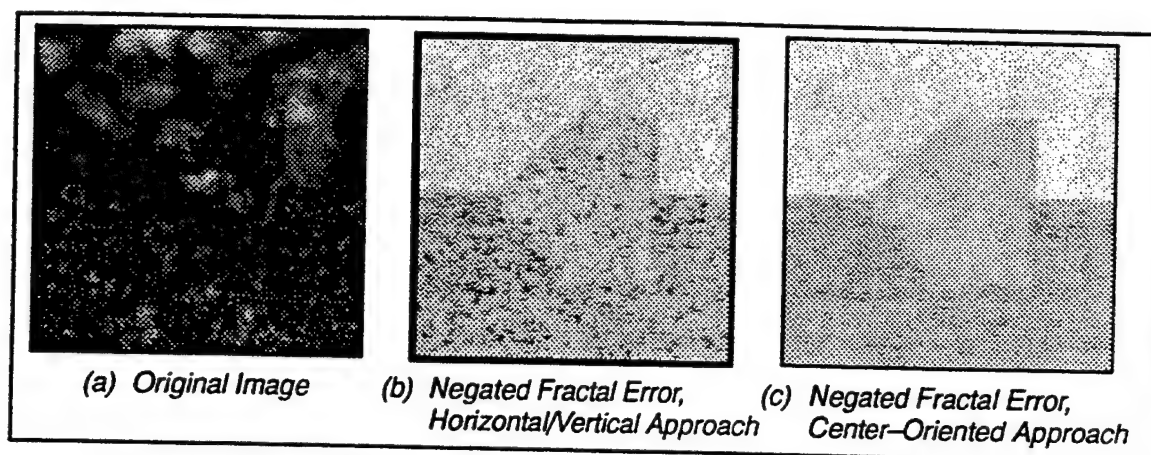
(a) Original Image   (b) Negated Fractal Error,   (c) Negated Fractal Error,
                          Horizontal/Vertical Approach   Center–Oriented Approach

Figure 55 Fractal Error Images of the Fractal Collage

appears dark. Both algorithms separated the regions; although the center–oriented
method produced less variance within the individual regions. This improvement is
apparent from the illustration.

Applied to aerial images, the center–oriented method was comparable or slightly
preferable to the horizontal and vertical approach. For example, the negated fractal error
images from each algorithm, using a 5×5 window size, are shown in Figures 56b
(normalized from the range [0.00, 14.48]) and Figure 56c (normalized from the range
[0.00, 17.27]) for the shore scene. The clarity of the center–oriented technique is better,
and the image is less spotty. Similar findings resulted for other aerial images. The RMS
error values from the horizontal and vertical method were always smaller, on average

(a) Original Image   (b) Negated Fractal Error,   (c) Negated Fractal Error,
                          Horizontal/Vertical Approach   Center–Oriented Approach
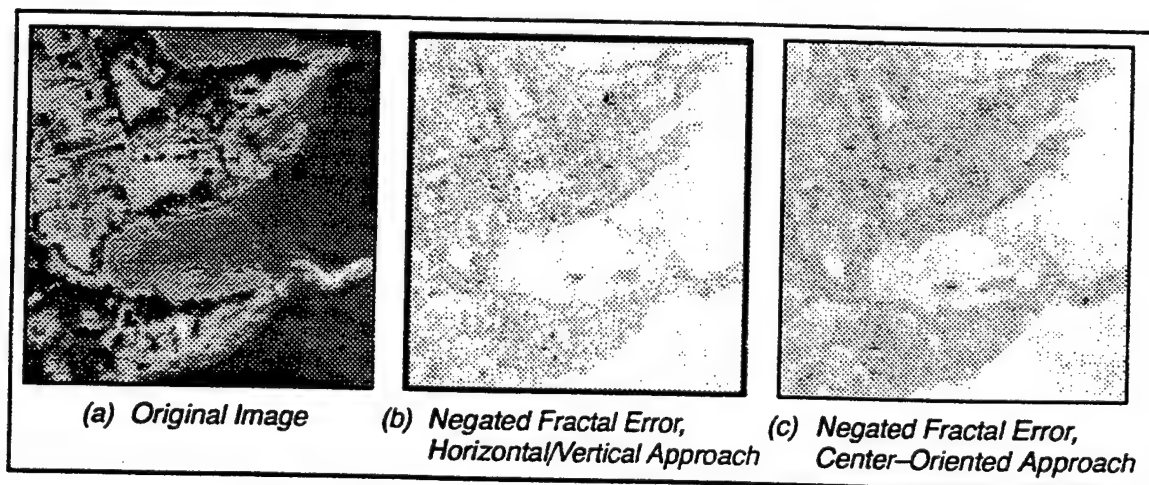
Figure 56 Fractal Error Images of the Shore Scene

about one third to one half of those from the center–oriented approach. Nevertheless, the visual quality of the center–oriented approach was superior, and it required less computational effort. Various sizes of windows were tried, but results favored the 5×5 size. Applications of the center–oriented fractal error measure, oriented toward aerial imagery, are explored in the following chapter.

CHAPTER VI

APPLICATIONS OF FRACTAL ERROR

6.1 Introduction

Focusing on aerial imagery, the fractal error was examined first as a generic texture feature. Originally, the fractal error was proposed as a potential alternative or supplement to the busyness operator, the texture feature previously chosen for classifying aerial images. The second application of fractal error considered here is the identification of cultural or man–made objects in an image. Because cultural objects are generally more stable than natural contents, such an ability would help to analyze the changes which occur in a scene over time.

6.2 Fractal Error as a Texture Feature

As part of the contract with the Naval Air Warfare Center to investigate texture features, a software package was designed to facilitate the segmentation of aerial imagery. The classification routines within this software use a combination of gray shade and texture to represent each pixel of the image. As described in section 5.1, the busyness operator was selected as the texture feature. It was quick, localized (using a 3×3 window) and general–purpose. The success of the fractal error operator on aerial imagery prompted an investigation to determine its value as a texture feature.

Described in section 5.5.3, the approach using center–oriented pixel distances within overlapping windows was selected as the fractal error operator. (Unless stated otherwise, a 5×5 window will be used.) The similarity between the images produced by the new fractal error feature and the existing busyness feature was apparent. For example, Figure 57 compares the fractal error and busyness images of the shore scene

(a) Fractal Error Image, Negated   *(b) Busyness Image, Negated*
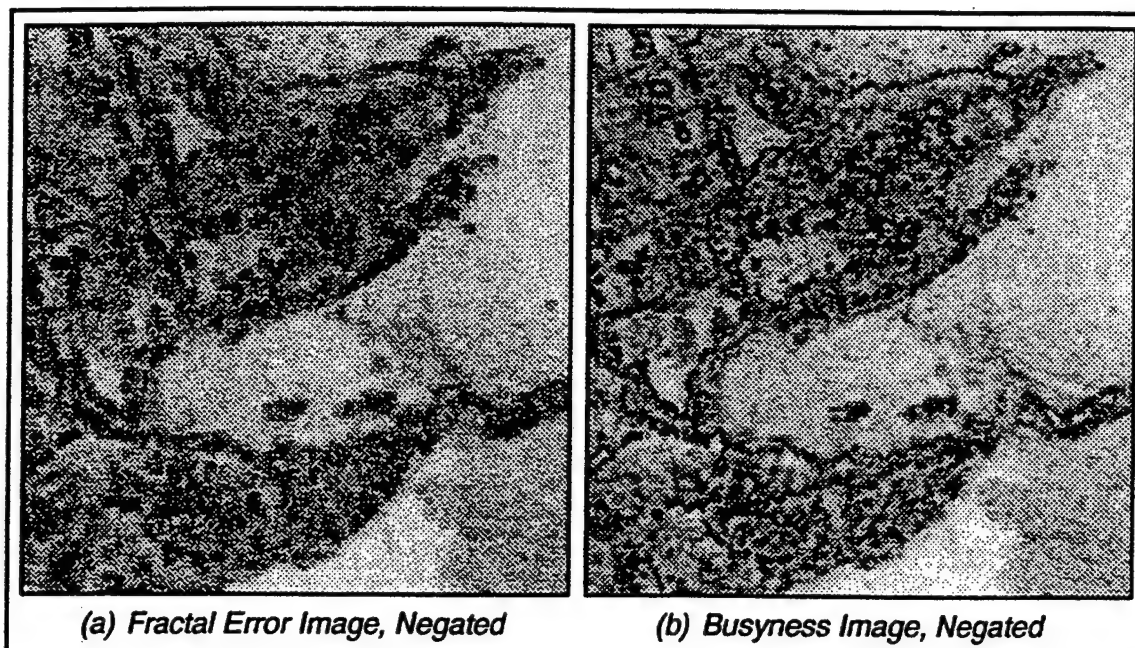
Figure 57   Comparison of Fractal Error and Busyness for the Shore Scene

(Figure 58a). To provide a clearer visual interpretation, both images were negated, and the gray shade distributions were enhanced to increase the contrast. The resemblance between the images is strong. Pairs of fractal error and busyness images were compared for a set of twelve aerial images which included a variety of natural and man-made content. In all cases, the results were similar. Upon closer examination, two primary differences were apparent. To its advantage, the fractal error images did not contain the sharp "specks" which often appeared in the busyness images. However, the fractal error highlighted abrupt transitions of gray shade that busyness sometimes neglected. (Recall from section 2.3.4.2 that the busyness operator will suppress edges which occur at 45 degree increments.)

Next, two-dimensional histograms of gray shade and fractal error were compared to those of gray shade and busyness. Figure 58b shows the histogram of gray shade and fractal error for the shore scene, displayed as an image. The darkness of each point indicates the relative number of occurrences for that particular pair of gray shade and fractal error values. Gray shade values range between zero and 255 on the horizontal axis, and fractal error values range from zero to 17.27 (the maximum fractal error

(a) Original Image

(b) Histogram

Foliage (left top)
Water (bottom center)
Land (right top)

Fractal Error

Gray Shade

(c) Classified Image

(d) Correct Segmentation
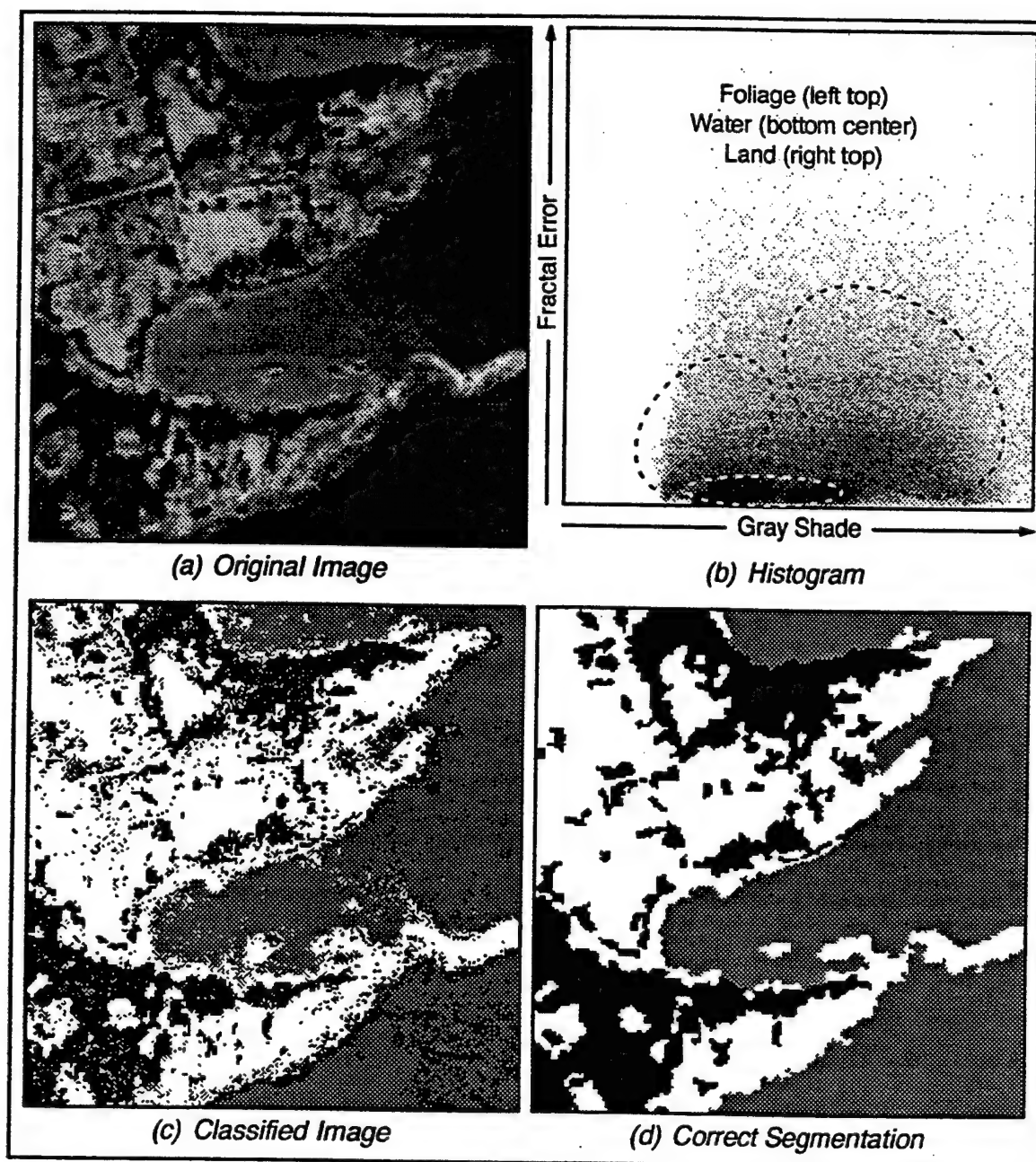
Figure 58 Fractal Error Applied to the Shore Scene

obtained for this image) on the vertical axis. The distributions of the three classes foliage

(trees), water and non–foliated land are approximated by the ellipses superimposed on

the histogram. Using a combination of gray shade and fractal error, these classes may be

separated, whereas either one of the features individually would fail. Of those

histograms examined, the clustering ability of gray shade and fractal error was

comparable to that of gray shade and busyness, as expected from the high degree of similarity between the fractal error and busyness images.

The fractal error was incorporated into the segmentation software in place of busyness, with favorable results. Figure 58c shows the classification of the shore scene into foliage (black), water (gray) and non–foliated land (white). This image was compared to the correctly segmented "ground–truth" image for these same three classes, shown in Figure 58d. Note that a certain degree of subjectivity is inevitable in the creation of such an image. In this case, no additional information about the contents of the scene was available. Even with such knowledge, human judgment would still be required, for example, to determine the subjective boundary between the beach and sea. In practice, a scene analyst relies on his experience to determine a suitable ground–truth segmentation. The image in Figure 58d was approved by such a trained scene analyst as a reasonable and accurate mapping of the ground features of the original shore scene in Figure 58a. Compared against this ground–truth image, the gray shade and fractal error classification gave an overall classification accuracy of 83.0%, whereas the classified image produced by the combination of gray shade and busyness (not shown) was only slightly more accurate, at 85.4%.

In addition to the shore scene, other aerial images were considered. (However, ground–truth imagery was unavailable for these images.) A scene of farm land is shown in Figure 59a. The light colored dirt areas are clearly distinguishable solely on the basis of gray shade intensity. However, the grassy fields and foliage require a texture measure to be separated. Figure 59b shows the negated fractal error image, in which the foliage tends to have a larger error (darker gray shade) than the fields. In Figure 59c, the image has been classified as foliage (dark gray), grass (light gray) or dirt (white). Finally, Figure 60a introduces a mixture of natural and cultural (e.g., roads, buildings) content. The lake in the upper left, the trees, the fields and cultural objects are generally separated in the negated fractal error image, shown in Figure 60b. The classification in Figure 60c
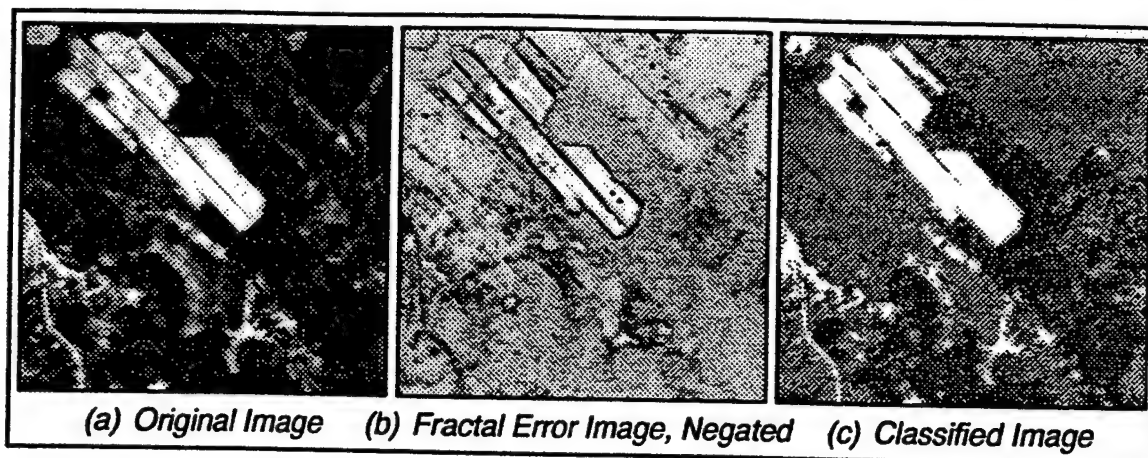
(a) Original Image   (b) Fractal Error Image, Negated   (c) Classified Image

Figure 59 Fractal Error Applied to the Farm Scene

(a) Original Image   (b) Fractal Error Image, Negated   (c) Classified Image
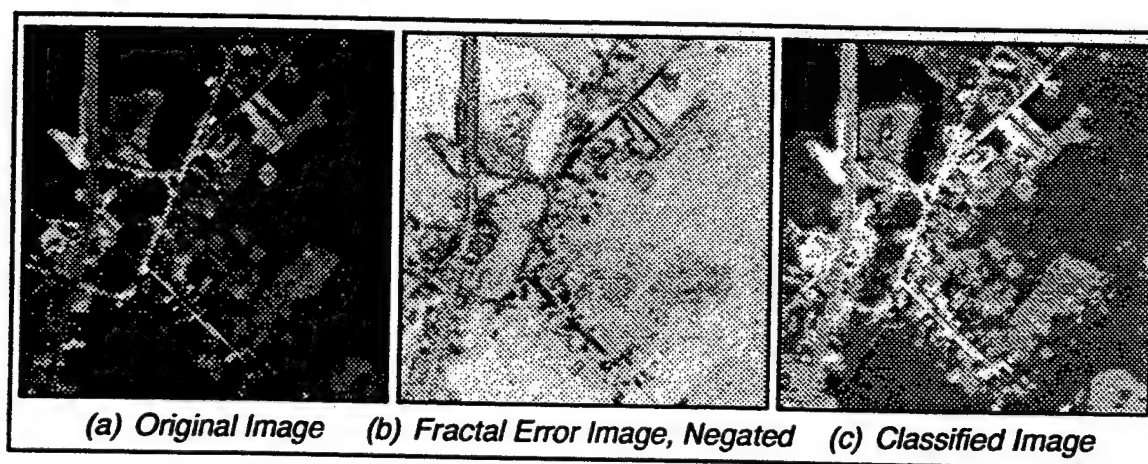
Figure 60 Fractal Error Applied to the Mixed Scene

identifies four classes, from dark to light: water, foliage, fields and cultural objects. As expected, classification using gray shade and fractal error was comparable to that obtained using gray shade and busyness.

The fractal error performed effectively as a natural texture feature, but the busyness gave similar results, in terms of the feature images, histograms and classifications. Since the quality of the two texture measures appeared equivalent, the deciding factor was computation time. The busyness feature is defined on a 3×3 neighborhood of pixels and uses simpler mathematics than the 5×5 definition of fractal error. Experimental results indicated that the busyness feature was approximately eight times faster than the fractal error. Although the fractal error produced favorable results,

the simpler texture feature, busyness, produced comparable results at a lower computational cost. Thus, busyness was retained as the texture feature in the segmentation software.

6.3 Fractal Error for Identification of Cultural Features

The fractal error should be large for regions which are not fractal. Since cultural or man–made objects often have a non–fractal appearance, the fractal error may have some potential use for aiding the identification of cultural features [7, 8]. Over an appropriate range of sizes, cultural objects will appear to be non–fractal. Clearly, a uniformly colored man–made object which is much larger than the window size will appear to be fractal in the center with a low fractal dimension corresponding to its smooth, uniform gray shade. Its boundary will very likely have a non–fractal appearance, since it represents a transition between two different textures. Thus, the correspondence between the object size and the window size used by the operator will be significant.

This property was examined in the context of two images of a residential area with a fairly high content of deciduous vegetation. The same scene was photographed in April and in October. This was done to evaluate the robustness of the metric relative to seasonal variations and changes in contrast and illumination that occurred at the different times. The primary man–made objects in the test images are about fifty to sixty pixels in size. The pair of images is shown in Figure 61. Figure 62 illustrates the fractal error of each of these images, using a 7 × 7 window size, thresholded so that features with large fractal errors appear black against a white background. It is obvious by comparing the original image and the threshold fractal error images that the fractal error performed reasonably well in detecting man–made features. It was also fairly insensitive to changes in illumination, contrast, and season. Note the brighter lighting and greater contrast in the
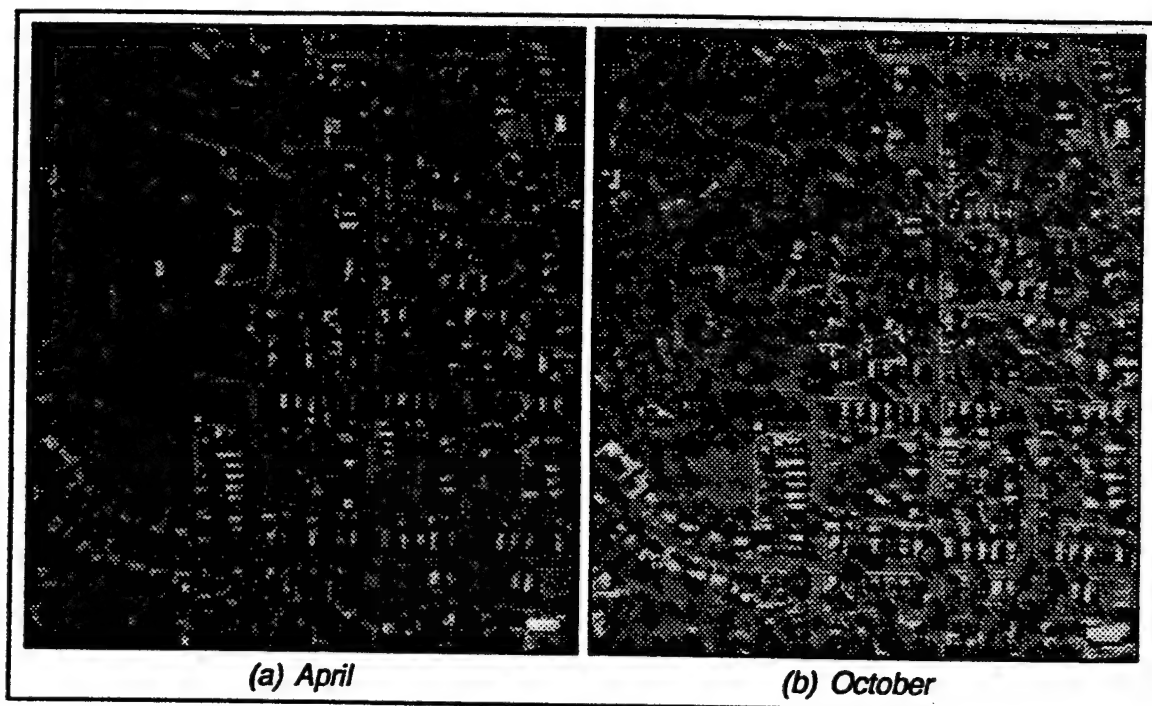
Figure 61 A Residential Scene, at Two Times of the Year
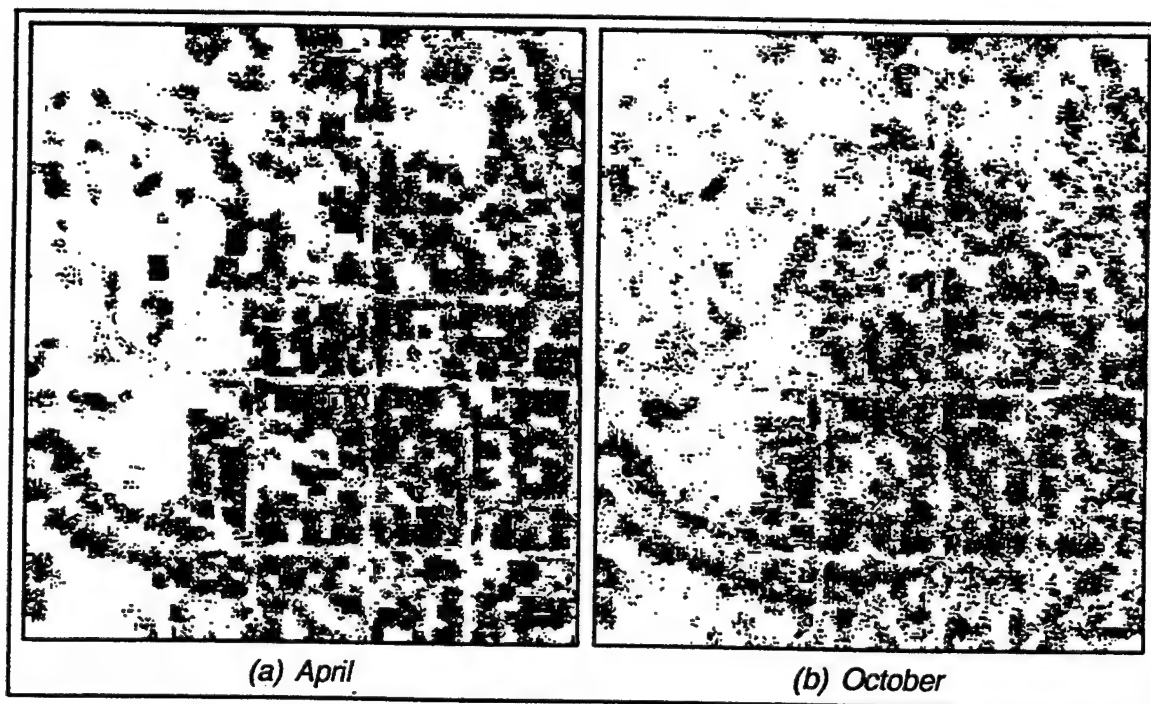
Figure 62 Binary Fractal Error Images of a Residential Scene

October image of Figure 61 compared to the April image, as well as the distinct increase of foliage in the October scene.

A combination of operators which could detect cultural features was tested on this pair of images. Cultural features usually possess clear edges, so three edge operators were included in the comparison. The fractal error with window sizes 5×5, 7×7 and 9×9 plus the busyness operator completed the set.

Quantifying an operator's accuracy for detecting the man–made content is difficult. As mentioned in section 6.2, the creation of a correctly segmented or "ground–truth" image involves a degree of subjectivity. Both the April and October images were used to create the ground–truth urban image shown in Figure 63 for the two classes "buildings" and "background." This image contains buildings which may have been obscured by foliage and shadows in one or possibly even both of the original images. For example, the presence of a rooftop in a shadow may sometimes be implied by the geometry of the structure. Likewise, a building with very low contrast might be detected by a faint shadow. In some circumstances, objects were blurry even at larger resolutions, requiring a subjective determination whether the particular item was a portion of a building. Although the images were aligned with one another (i.e.,

Figure 63 Correct Segmentation of Residential Scene

registered), there were slight misalignments of one or two pixels around the upper right and lower left corners of the image. The ground–truth segmentation attempted to compromise between the two images in such cases.

When comparing the fractal error, busyness and edge operators, each of the filtered images produced by these operators was thresholded manually, while attempting to maximize the distinction between buildings and background. Although this task was performed carefully, it is inherently subjective. A more reliable comparison might consider a collection of several binary images which use different thresholds.

The fractal error and busyness filtered images were compared directly to the ground–truth image. However, one would not expect the edge operators to match well to the filled objects of the correctly segmented image. At best, they can detect only the object boundaries, although this should be considered a valid identification of the object. Therefore, each edge operator was also applied to the ground–truth image, and the result was thresholded to a binary image. (While partially subjective, the determination of a threshold was much more obvious for this step than for the previous edge images.) Thus, the edge image of the ground–truth image was compared to the edge image of the original image for each of the three edge operators.

For each comparison, the percentage of correctly classified pixels was computed. The overall classification accuracy, based upon the relative number of pixels labeled as either buildings or background, was considered. However, this value can be misleading here, since there were about five times more pixels labeled as background than as buildings. (For example, an image consisting entirely of background would still manage to produce an excellent overall classification accuracy above 80%.) Since the goal is the detection of buildings, the percentage of buildings classified as buildings might be a better means of comparison, although one could obtain perfect results with an image consisting entirely of buildings. However, the images were thresholded to produce meaningful results, not pathological situations, so this percentage may still be useful. An

alternative criterion is the simple (non–weighted) average of the two matching classifications (i.e., buildings classified as buildings and background classified as background). Here, each the buildings and background classes are given equal importance.

The fractal error correctly classified around 60% to 70% of the buildings, compared to around 50% to 60% for the busyness and 10% to 35% for the edge operators. The average classification accuracy was roughly equal for the fractal error and busyness, between 60% and 70%. This measure dropped to about 50% to 60% for the edge operators. A more thorough comparison might consider a more sophisticated decision criterion and a collection of binary filtered images obtained with various thresholds. Nevertheless, these results indicate that the visual appeal of the fractal error for identifying cultural features has quantitative support.

This analysis led to another important task in the evaluation of aerial imagery, the enhancement of those objects in a scene which are stable over a period of time. Typically, cultural features are more stable than natural features. It was also desired that the resulting pair of enhanced images should be very similar, indicating their common origin. The same combination of operators which was considered previously for detecting cultural features was tested on the previous pair of images: three edge operators, the busyness operator and the fractal error with window sizes 5×5, 7×7 and 9×9.

For each of the seven operators, the following test was conducted. (Recall that the previous test verified that the images were properly aligned with one another.) The pair of images was processed by the given operator, producing a pair of filtered images. Subtracting one filtered image from the other yielded the difference image. Ideally, the difference image should be uniformly zero: zero mean and zero variance. The root mean square (RMS) error was chosen as the statistic to characterize the difference image. Equal to the square root of the sum of the variance and squared mean, the RMS error

incorporates both of these statistics together into a single measure. (Note that the mean and variance of either the ordinary difference or absolute difference may be used; the result is mathematically equivalent.) However, the RMS error from two different operators cannot be compared directly, since the signal strengths of the filtered images may differ between operators. (For example, consider two signals with a given mean value. Scaling the signals by a constant factor will also scale their difference and RMS error by the same constant.) To compensate for differing signal strengths, each RMS error was normalized by the average magnitude of its particular filtered images. More precisely, suppose that μ_1 and μ_2 are the average values for a pair of feature images. Then the normalized RMS error will be as shown below:

$$\text{normalized RMS error} = \frac{\text{RMS error}}{(\mu_1 + \mu_2)/2}$$

Figure 64a lists the normalized RMS error values for the seven features, ranked in order. (Each operator's window size is also indicated.) The 7×7 and 9×9 fractal error gave the best results. Unfortunately, the 5×5 fractal error did not perform as well as expected, trailing busyness and leading the Sobel edge operator by a margin that was not especially large. Although they clearly identified the edges of cultural objects, the edge operators generally performed poorly because they also located the edges of unstable items, such as shadows. Another pair of images, of a different residential area,

Operator	Normalized RMS Error	Operator	Normalized RMS Error
Fractal Error 9x9	0.643	Fractal Error 9x9	0.529
Fractal Error 7x7	0.734	Fractal Error 7x7	0.624
Busyness 3x3	0.907	Sobel 3x3	0.801
Fractal Error 5x5	0.937	Busyness 3x3	0.807
Sobel 3x3	0.968	Fractal Error 5x5	0.833
Prewitt 3x3	1.205	Prewitt 3x3	1.013
Laplacian	1.320	Laplacian	1.184
(a) Original Pair of Residential Scenes		(b) Second Pair of Residential Scenes	

Figure 64 Comparison of Operators for Cultural Feature Stability

was also examined. The resolution was approximately the same as before. One image was taken in April and the other in October. Listed in Figure 64b, these results also indicate that the 7×7 and 9×9 fractal error operators were the most suitable for detecting the stable contents of the scene.

CHAPTER VII

OBSERVATIONS AND CONCLUSIONS

The primary purpose of this research was to determine fractal characteristics suitable for representing visual textures, and to develop algorithms to extract such characteristics, with particular emphasis on aerial imagery. Under the original problem statement (section 1.4), the following questions were proposed:

- How appropriate is the fractional Brownian motion model for visual texture?

- What are some useful fBm synthesis algorithms and how do they compare?

- What are some useful fractal characteristics, how may they be estimated, and how do their extraction algorithms compare?

- How can estimation algorithms be applied?

- How may the fractal characteristics be used within a knowledge–based segmentation system?

The first question will conclude this discussion. The other questions will be addressed in order.

Artificial fractional Brownian motion (fBm) images were created in three different manners. The spectral synthetic algorithm relies upon the frequency distribution of fBm. The other two spatial synthesis approaches use the random midpoint displacement algorithm. One included successive random additions while the other did not. In all cases, the visual roughness of the synthetic images increased along with their fractal dimension. Although artificially generated, these textures appeared very natural. However, only the spectral method is theoretically sound. Detailed analysis revealed that the random midpoint displacement algorithm preserved the properties of fBm only for certain pixel spacings. Furthermore, the practice of including successive random

additions to remove the creasing effect actually decreases the theoretical validity of the algorithm. Nevertheless, all three synthesis algorithms were included in the subsequent estimation of fractal characteristics. The estimates of the random midpoint displacement fractals were made to produce a more complete study and to provide insights from an algorithm whose empirical analysis is absent in the known literature. Although the random midpoint displacement algorithm was not studied in detail, there were no trends between its theoretical problems and its measured fractal properties.

The most obvious and well–studied fractal characteristic is the dimension. The fractal dimension corresponds directly to the perceptual notion of visual roughness. Few researchers have considered the error in their estimation. Thus, the degree to which the measured signal fits the fBm model was included in this study. Also considered was the lacunarity, which seems to be related to the coarseness or granularity of the texture. However, formulations for estimating the lacunarity suffer from a lack of consistency, and very little work exists on synthesizing textures with a particular lacunarity value. Therefore, the fractal dimension and the fractal error were the two characteristics explored here.

Before estimating the fractal dimension and fractal error of a texture, the fractal scaling property was considered, which requires an exponential relationship between the signal spacing and the average change in signal strength. The synthetic fractals adhered closely to this requirement, especially the spatially synthesized fractals. The Brodatz textures deviated from this property, as expected, with generally reasonable results. The "clouds," "sand" and "waves" textures matched the best, while the "burlap" and "paper" textures fit the worst. Next, three methods were applied to the estimation of the fractal dimension and fractal error across the entire image: spectral estimation, spatial estimation and box–counting. It was anticipated that the theoretical inconsistencies in the spatial synthesis algorithms might affect the accuracy of their estimates. For example, the spectral approach gave close estimates for spectrally synthesized fractals, but the

values for the spatial fractals were consistently high. However, the accuracy of the spatial technique was very close for all sets of artificial fractals. In addition to its consistency, the spatial estimation algorithm gave the most accurate results also. Its accuracy was close to that produced by the improved interpolated box–counting approach presented by Keller et al. [25]

Although the attributes of a fractal should remain invariant across any resolution, findings indicated that the dimension and error were not necessarily constant. The estimated dimension curves remained fairly flat for the spatial fractals, but they increased slightly for the spectral fractals. The fractal RMS error decreased rapidly up to a distance of about four to six pixels before leveling out. As expected, the linearity of the dimension and error of the Brodatz textures varied, but the RMS error was always higher, on average six times greater than the RMS error for the synthetic fractals.

For segmentation, the measure must be restricted to a small, local portion of the image. Statistics were collected on the synthetic fractals for each estimation algorithm, yielding surprisingly accurate dimension estimates compared to the global algorithms. Unfortunately, the dimension images of each spectral, horizontal/vertical and center–oriented local estimation were of poor quality. However, the fractal error proved to be a useful texture feature. Efforts focused on the faster center–oriented estimation, which offered better quality.

Fractal error was examined as a texture feature, specifically for segmenting aerial images. In conjunction with gray shade, the fractal error could distinguish the regions in several aerial images effectively. It bore a strong resemblance to the texture feature previously chosen for aerial image segmentation, the busyness operator. However, because the fractal error took longer to compute, busyness was retained as the texture feature in the aerial image segmentation software.

Noting that cultural or man–made objects in an image will typically be non–fractal, the fractal error was applied to the identification of cultural features. The

cultural content of the scene was located successfully using fractal error. Tested against other simple operators which could detect cultural features, the 9×9 and 7×7 window sizes of fractal error were found to perform best. These operators were least affected by changes in illumination, contrast and season.

The notion of modeling texture with fBm originated from the work of Pentland [39]. The justification stems from the assertion that natural terrain is created and shaped by fractal processes and is supported by the convincing appearance of fBm renderings of terrain surfaces [30]. Although an image intensity surface of a fBm surface will also produce fBm, the reflectance in the original surface will usually vary. Under such typical conditions, the argument that an image is inherently a fBm signal lacks rigor. Nevertheless, solid evidence exists linking the fractal dimension of a fBm surface with its perceived visual roughness. Because roughness is only one aspect of texture, a pair of textures could have the same roughness but otherwise be quite distinct. Similarly, the Brodatz textures "clouds" and "fieldstone" are noticeably different, although their estimated fractal dimensions were nearly identical.

Dimension estimates which considered larger image regions produced accuracies which were competitive with other published work and sometimes even better. Whereas most researchers have estimated the dimension within fairly large regions, a much smaller size was required for the resolution of the aerial imagery in this study. Such a restriction decreased the impact of the fractal dimension estimates. However, the fractal error proved to be a viable feature, even when estimated within very small regions of the image. Even for larger regions, the degree to which the actual image fits the model is highly significant. However, very few researchers have considered this error in their estimations.

The following areas of further research are proposed. First, the effects of quantizing the signal into only 256 values should be examined. Even more than sampling the continuous fBm signal into a discrete signal, the quantization is suspected to

be responsible for at least some of the inaccuracies in the estimated fractal dimension. For example, the global spectral estimates of the globally synthesized fractals should be highly accurate, since both share the same theoretical foundation. Brief experiments using a larger range of signal values indicated promise.

Although the theory predicts that fractal characteristics should remain invariant over all resolutions, empirical findings revealed that this was not completely true. These effects could result from flaws in the synthesis, or they could be related to the estimation process. The findings could be used to develop more reliable synthesis and estimation algorithms.

In addition to furthering the research on fractal dimension and fractal error, the lacunarity and other potential characteristics should be investigated. Just as dimension corresponds to roughness, lacunarity appears to describe the coarseness or granularity of a texture. A successful representation of roughness and coarseness, both of which relate inherently to fractal geometry, would provide a more detailed description of visual texture.

Looking further ahead, the relationship between fBm and other texture models, such as co-occurrence matrices, Fourier spectra, Markov processes and autocorrelation deserves closer attention. In particular, parameters which are difficult to estimate in one paradigm might benefit from a potential link to more easily estimated parameters in another paradigm.

In another area, the fractal model appears to be naturally suited toward multiple resolution analysis. In simplest terms, fractal texture analysis may be applied to a hierarchy of image resolutions. At a deeper level, the inherent scaling properties of self–affinity provide a stronger link to the concept of a representation which spans a range of resolutions. Also, the correspondence between fBm and Markov random fields serves as additional indication that such work deserves further attention.

APPENDIX

Given a normally distributed random variable X with a mean of zero and variance σ^2, the mean of $|X|$ will be proportional to σ.

1. Let the probability distribution of X be labeled $f(x)$. From the given information,

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(\frac{-x^2}{2\sigma^2}\right)$$

2. Let $Y = |X|$. Find the probability distribution $g(y)$ of Y. (Note that $G(y)$ and $F(x)$ denote the cumulative distributions of $g(y)$ and $f(x)$, respectively.)

$$G(y) = P(Y \le y) = P(|X| \le y) = P(-y \le X \le y) = F(y) - F(-y)$$

Differentiating with respect to y gives $g(y) = f(y) + f(-y)$.

Since y must be non-negative, $g(y) = 0$ for $y < 0$.

The distribution of $f(x)$ is symmetric about the origin, so

$g(y) = 2f(y)$ for $y \ge 0$.

Thus,

$$g(y) = \frac{2}{\sigma\sqrt{2\pi}}\exp\left(\frac{-y^2}{2\sigma^2}\right) \text{ for } y \ge 0$$

3. Find the mean of Y. By definition,

$$E[Y] = \int_{-\infty}^{\infty} y\, g(y)\, dy = \frac{2}{\sigma\sqrt{2\pi}}\int_{0}^{\infty} y\, \exp\left(\frac{-y^2}{2\sigma^2}\right) dy =$$

$$\frac{-2\sigma^2}{\sigma\sqrt{2\pi}}\int_{0}^{\infty} \exp\left(\frac{-y^2}{2\sigma^2}\right)\frac{-y}{\sigma^2}\, dy = \frac{-\sqrt{2}\sigma}{\sqrt{\pi}}\exp\left(\frac{-y^2}{2\sigma^2}\right)\Bigg|_{0}^{\infty}$$

Since the limit at infinity is zero, this becomes

$$E[Y] = E[|X|] = \sigma\sqrt{\frac{2}{\pi}}$$

4. Thus, the mean of $|X|$ is proportional to the standard deviation of X, where the proportionality constant is $\sqrt{2/\pi}$.

REFERENCES

1. Ballard, D.H. and Brown, C.M., *Computer Vision*, Englewood Cliffs, NJ: Prentice–Hall, 1982.

2. Barnsley, M.F., *Fractals Everywhere*, Boston: Academic Press, Inc., 1988.

3. Brodatz, P., *A Photographic Album for Artists and Designers*, New York: Dover, 1966.

4. Conners, R.W., Trivedi, M.M. and Harlow, C.A., "Segmentation of a High–Resolution Urban Scene Using Texture Operators," *Computer Vision, Graphics, and Image Processing*, Vol. 25, pp. 273–310, 1984.

5. Cooper, B.E., "The Gray Shade and Texture Segmentation Software User Manual," Technical Report, Louisville, KY: University of Louisville, October 1991.

6. Cooper, B.E., "Gray Shade and Texture Segmentation Software Development for Segmenting Aerial Reconnaissance Images," Technical Report, Louisville, KY: University of Louisville, March 1992.

7. Cooper, B.E. and Chenoweth, D.L., "On the Use of Fractals for Interpreting Aerial Reconnaissance Images," *Thirteenth GRETSI Symposium on Signal and Image Processing*, Vol. 2, pp. 1141–1144, 16–20 Sept. 1991.

8. Cooper, B.E., Chenoweth, D.L. and Selvage, J.E., "Fractal Error for Detecting Man–Made Features in Aerial Images," *Electronics Letters*, Vol. 30, No. 7, pp. 554–5, 31 March 1994.

9. Cross, G.R. and Jain, A.K., "Markov Random Field Texture Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI–5, No. 1, pp. 25–39, January 1983.

10. Dauben, J.W., *Georg Cantor: His Mathematics and Philosophy of the Infinite*, Cambridge, MA: Harvard University Press, 1979.

11. Dennis, T.J. and Dessipris, N.G., "Fractal Modelling in Image Texture Analysis," *IEE Proceedings*, Vol. 136, Part F, No. 5, pp. 227–235, October 1989.

12. Derin, H. and Cole, W.S., "Segmentation of Textured Images Using Gibbs Random Fields," *Computer Vision, Graphics, and Image Processing*, Vol. 35, pp. 72–98, 1986.

13. Falconer, K., *Fractal Geometry – Mathematical Foundations and Applications*, New York: John Wiley and Sons, Ltd., 1990.

14. Fournier, A., Fussell, D. and Carpenter, L., "Computer Rendering of Stochastic Models," *Communications of the ACM*, Vol. 25, No. 6, pp. 371–384, June 1982.

15. Gonzalez, R.C. and Wintz, P., *Digital Image Processing*, Reading, MA: Addison–Wesley, 1987.

16. Hall, E.L., *Computer Image Processing and Recognition*, New York: Academic Press, 1979.

17. Hanson, A. and Riseman, E., "The VISIONS Image–Understanding System," *Advances in Computer Vision*, Vol. 1, ed. C. Brown, Hillsdale, NJ: Lawrence Erlbaum Associates, 1988, pp. 1–160.

18. Haralick, R.M., "Statistical and Structural Approaches to Texture," *Proceedings of the IEEE*, Vol. 67, No. 5, pp. 786–804, May 1979.

19. Haralick, R.M., Shanmugam, K. and Dinstein, I., "Textural Features for Image Classification," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC–3, pp. 610–621, Nov. 1973.

20. Hawkins, J.K., "Textural Properties for Pattern Recognition," *Picture Processing and Psychopictorics*, eds. B.S. Lipkin and A. Rosenfeld, New York: Academic Press, pp. 347–370, 1970.

21. Hsiao, J.Y. and Sawchuck, A.A., "Unsupervised Textured Image Segmentation Using Feature Smoothing and Probabilistic Relaxation Techniques," *Computer Vision, Graphics, and Image Processing*, Vol. 48, pp. 1–21, 1989.

22. Kaizer, H., "A Quantification of Textures on Aerial Photographs," Technical Note 121, AD 69484, Boston, MA: Boston University Research Laboratories, 1955, cited by Haralick, R.M., "Statistical and Structural Approaches to Texture," *Proceedings of the IEEE*, Vol. 67, No. 5, pp. 786–804, May 1979.

23. Kaneko, H., "A Generalized Fractal Dimension and Its Application to Texture Analysis," *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1711–1714, 23–26 May 1989.

24. Keller, J.M., Crownover, R.M. and Chen, R.Y., "Characteristics of Natural Scenes Related to the Fractal Dimension," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI–9, No. 5, pp. 621–627, September 1987.

25. Keller, J.M., Chen, S. and Crownover, R.M., "Texture Description and Segmentation through Fractal Geometry," *Computer Vision, Graphics, and Image Processing*, Vol. 45, pp. 150–166, 1989.

26. Kirvida, L., "Texture Measurements for the Automatic Classification of Imagery," *IEEE Transactions on Electromagnetic Compatibility*, Vol. 18, pp. 38–42, February 1976.

27. La Brecque, M., "Fractal Applications," *Mosaic*, Vol. 17, No. 4, pp. 34–48, Winter 1986/7.

28. Laws, K.I., *Textured Image Segmentation*, Ph.D. dissertation, Department of Engineering, University of Southern California, Los Angeles, CA, 1980.

29. Lendaris, G.G. and Stanley, G.L., "Diffraction–Pattern Sampling for Automatic Pattern Recognition," *Proceedings of the IEEE*, Vol. 58, No. 2, pp. 198–216, February 1970.

30. Mandelbrot, B.B., "Stochastic models for the Earth's relief, the Shape and the Fractal Dimension of the Coastlines, and the Number–Area Rule for Islands," *Proceedings of the National Academy of Sciences USA*, Vol. 72, pp. 3825–3828, 1975.

31. Mandelbrot, B.B., *Fractals: Form, Chance and Dimension*, San Francisco: W.H. Freeman and Co., 1977.

32. Mandelbrot, B.B., "Comment on Computer Rendering of Stochastic Models," *Communications of the ACM*, Vol. 25, No. 8, pp. 581–584, August 1982.

33. Mandelbrot, B.B., *The Fractal Geometry of Nature*, New York: W.H. Freeman and Co., 1983.

34. Mandelbrot, B.B. and Van Ness, J.W., "Fractional Brownian Motions, Fractional Noises and Applications," *SIAM Review*, Vol. 10, No. 4, pp. 422–437, October 1968.

35. Ohta, Y., *Knowledge–based Interpretation of Outdoor Natural Color Scenes*, Boston: Pitman Advanced Publishing Program, 1985.

36. Papoulis, A., *Probability, Random Variables, and Stochastic Processes*, New York: McGraw–Hill, Inc., 1991.

37. Peitgen, H.–O., Jürgens, H. and Saupe, D., *Chaos and Fractals: New Frontiers of Science*, New York: Springer–Verlag, 1992.

38. Peitgen, H.–O. and Saupe, D. (eds.), *The Science of Fractal Images*, New York: Springer–Verlag, 1988.

39. Pentland, A.P., "Fractal–Based Description of Natural Scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI–6, No. 6, pp. 661–674, November 1984.

40. Pickett, R.M., "Visual Analyses of Texture in the Detection and Recognition of Objects," *Picture Processing and Psychopictorics*, eds. B.S. Lipkin and A. Rosenfeld, New York: Academic Press, pp. 289–308, 1970.

41. Pratt, W.K., *Digital Image Processing*, New York: John Wiley & Sons, Inc., 1978.

42. Pratt, W.K., Faugeras, O.D. and Gagalowicz, A., "Applications of Stochastic Texture Field Models to Image Processing," *Proceedings of the IEEE*, Vol. 69, No. 5, pp. 542–551, May 1981.

43. Rosenfeld, A. and Kak, A., *Digital Picture Processing*, Vol. 2, Academic Press, 1982.

44. Rosenfeld, A. and Troy, E.B., "Visual Texture Analysis," Proceedings UMR–Mervin J. Kelly Communications Conference, University of Missouri–Rolla, Missouri, October 1970, Section 10–1, cited by Pratt, W.K., *Digital Image Processing*, New York: John Wiley & Sons, 1978.

45. Trivedi, M.M. and Harlow, C.A., "Identification of Unique Objects in High Resolution Aerial Images," *Optical Engineering*, Vol. 24, No. 3, pp. 502–506, May/June 1985.

46. Vehel, J.L., "Using Fractal and Morphological Criteria for Automatic Classification of Lung Diseases," *Visual Communications and Image Processing IV*, SPIE Vol. 1199, pp. 903–912, 8–10 November 1989.

47. Voss, R.F., "Fourier Synthesis of Gaussian Fractals: 1/f Noises, Landscapes, and Flakes," in *State of the Art in Image Synthesis*, Tutorial No. 10, SIGGRAPH 1983.

48. Voss, R.F., "Random Fractals: Characterization and Measurement," *Scaling Phenomena in Disordered Systems*, eds. R. Pynn and A. Skjeltorp, New York: Plenum Press, pp. 1–11, 1985.

VITA

Brian Edward Cooper was born on April 4, 1965 in Covington, Kentucky, to Joseph Leslie and Martha Jane Cooper, as the youngest of their three children.

Upon graduating first in his class from Saint Xavier High School in Louisville, he enrolled in the Speed Scientific School at the University of Louisville as a National Merit Finalist. During this time, he served as student chapter president of the Association for Computing Machinery and student arrangements chair for the 1987 Tau Beta Pi Engineering Honor Society's national convention. After completing his Bachelor of Science degree with highest honors in December 1986, he earned his Master of Engineering degree, also with highest honors, in May 1989.

He taught for the Engineering Math and Computer Science Department at the University of Louisville before accepting a position as Research Associate in 1990. He expects to receive his Doctorate in Computer Science and Engineering in August 1994. Beyond his technical interests in image processing, fractals and artificial intelligence, Brian enjoys hiking, volleyball and bicycling.

FEATURE EXTRACTION AND EDGE DETECTION USING A FRACTAL-BASED METRIC

By

E. David Jansing
B.S.E.S., University of Louisville, 1991
M.Eng., University of Louisville, 1992

A Dissertation
Submitted to the Faculty of the
Graduate School of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

.

Computer Science and Engineering Program
University of Louisville
Louisville, Kentucky

December 1997

FEATURE EXTRACTION AND EDGE DETECTION USING A FRACTAL-BASED METRIC

Submitted by

E. David Jansing

A Dissertation Approved on

November 21, 1997
(Date)
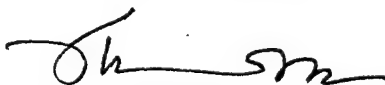
by the Following Reading and Examination Committee:
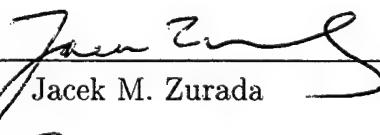
_____
Darrel L. Chenoweth
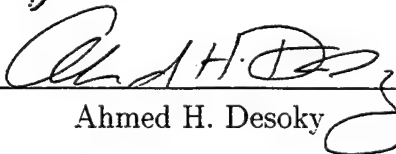
_____
Melvin J. Maron

_____
Prasanna K. Sahoo

_____
Jacek M. Zurada

_____
Ahmed H. Desoky

i

Dedicated to

my grandfather

Michael B. Hagan

– a man with infinite imagination!

ACKNOWLEDGMENTS

ABSTRACT

Fractal error is an image processing metric that can be used to locate man-made features in aerial images, as well as edges in industrial images. The metric can aid photo interpreters in locating targets in aerial reconnaissance images. Since the development of fractal error, it has been shown that the fractal error metric also works well for extracting features in synthetic aperture radar (SAR) images. A novel method is presented for automatic 2-D entropic segmentation using fractal error as a feature.

Previous work has also shown that the fractal error metric is useful for locating edge pixels in industrial images. An introduction to edge detection using fractal error is presented; the results of the fractal error edge detection algorithm are compared with the Canny edge operator for robustness and accuracy.

Fractal error has many useful applications; however, some applications require real-time image analysis. The main disadvantage of the fractal error algorithm is that it can take several seconds to compute on large images. Therefore, it is desirable to create an approximation of fractal error to provide real-time image analysis. A novel approximation of fractal error using a genetic algorithm is also presented.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I
INTRODUCTION

The mathematical nature of textures within images has generated much interest in the literature in past years. Specifically, fractal dimension and lacunarity are commonly the focus of such interest. Notably, Pentland [49] used fractal dimension to analyze natural visual textures. However, little attention has been given to measuring the fitness of the fractal model itself. Cooper [12], [13] conducted the first research indicating that there may be some use in a measure that quantifies the error between the calculated fractal model and the actual data. This study is an extension of Cooper's work. The practical use of the fractal error as a feature measure, both in segmentation and edge detection, is explored here.

A.   Major Contribution

Upon examination, there are three important examples that can be described as motivation for this research. The first example involves segmentation of cultural objects in synthetic aperture radar (SAR) imagery. Reliable edge detection in the presence of noise is the second. An approximation to fractal error for real-time processing is the third.

For example, mission planning is a critical part of a tactical aircraft mission. To ensure the safety of the pilot and the success of the mission, extensive planning is required. Greater efficiency and consistency are achieved by supplementing pilot calculations with machine automation and intelligence. However, during the actual execution of the mission, the tactical situation may undergo unpredictable change.

Instead of returning to the aircraft carrier and re-planning a new mission, it may be desirable to modify the mission en route. Therefore, it is necessary to data link mission-related information to aid the pilot with his new objective. Surveillance and reconnaissance imagery coupled with other intelligence information and digital maps can be presented to the flight crew. Included in this mission-related data may be processed imagery of the target area from off-board sensors. These off-board sensors may provide infrared (IR) imagery, SAR imagery, and optical imagery. SAR imagery is a likely sensor candidate. While highly specular and often difficult to interpret without enhancement, SAR is less susceptible to weather conditions and ambient illumination, than IR. Thus, from all of the data sent to the flight crew, terrain features, line-of-sight perspectives, and expected target area features can be displayed in order to highlight the new mission aspects. The fractal error metric is described here in a method that provides useful detection of cultural objects. The resulting imagery could be used in on-the-fly mission adjustments.

As another example, edges typically do not fit the fractional Brownian motion (fBm) model well, due to their irregularity. Therefore, fractal error is also a practical edge enhancement operator. In this study, an algorithm has been developed, using fractal error as a likelihood function, to separate the edges from the other features in images. Edge linking, or the process of making discontinuous edges continuous, is the second stage in the edge detection process. A new method of edge linking using fractal error is introduced.

Many applications require real-time processing. For example, automatic target recognition is worthless if several minutes are dedicated to processing the image. It will be shown that fractal error is a computationally intensive algorithm. Thus, a fast approximation to fractal error is desirable. This work will describe a genetic approximation to fractal error. A genetic algorithm is used to evolve a set of weights in a fixed mathematical structure. The structure, along with the final weights, are

then used to approximate fractal error.

This dissertation will explore the functionality of fractal error, the genetic approximation of the algorithm, and a method of edge detection using the fractal error metric. Chapter II will briefly review the literature on fractal and texture analysis and edge detection. Chapter III will present an introduction to fractal error. Chapter IV will describe the approximation to the fractal error metric using evolutionary computing, specifically, genetic algorithms. The issue of automatic thresholding for cultural object detection will be addressed in Chapter V, which outlines a method using a 2-D entropy decision line to segment the image. Edge enhancement and edge detection will be examined in Chapter VI. Each chapter will present results, the advantages and disadvantages of each developed method and future research ideas.

CHAPTER II
PREVIOUS WORK

A.   Feature Detection and Synthetic Aperture Radar

There has been much work presented in the literature regarding feature extraction, segmentation, and object recognition with respect to synthetic aperture radar (SAR) images. Many of the articles summarized in this section deal with terrain images and few attempt to address the problem of recognition of man-made objects in the images.

For example, He and Wang [27] presented a new set of textural measures derived from the texture spectrum and applied these measures to SAR imagery. The measures developed in their study extract textural information of an image. These measures provide information from the eight directions in a localized 3x3 neighborhood in the image.

The basis of the features suggested by He and Wang is that a texture image will be considered as a set of texture units which characterize the local texture information for a given pixel and its neighborhood. From these local texture units, an image of the global texture aspects can be mapped. Using the local texture units, the frequency distribution of the texture units is calculated, with the abscissa representing the texture unit number and the ordinate representing its occurrence frequency. From this frequency distribution comes the term texture spectrum. He and Wang use this texture spectrum to extract information to form quantitative feature vectors which incorporate characteristics of the texture, such as black-white symmetry, geometric

symmetry, degree of direction, orientational features and central symmetry. He and Wang compared their results to previous research using the co-occurrence matrix approach [34], [20], [23], [11], and [26]. The results indicate that the He and Wang measures provide a more detailed feature vector.

Other authors have suggested using statistical image models as a means to segment SAR data [15], [38], [57]. In an article by Rignot and Chellappa [52], a model for the conditional distribution of the polarimetric complex data is combined with a Markov random field representation in order to segment the SAR image. Their article describes a model for the posterior distribution of the region labels. Optimization of those region labels is defined as maximizing the posterior distribution of the region labels, or maximum *a posteriori* (MAP) estimate.

Solberg, Taxt and Jain [57] developed a general model for multisource classification of remotely sensed data, such as optical, IR and SAR. Their model is based on Markov Random Fields (MRF) and was developed for the fusion of optical images, SAR images and Geographic Information Systems (GIS) ground cover data. Their model uses the spatial class dependencies, also called spatial context, between pixel neighbors in an image and temporal class dependencies between different images of the same scene. They have tested their model on Landsat TM images, ERS-1 SAR images and GIS ground-cover maps on agricultural scenes with encouraging results.

Geometric feature extraction techniques have also been used to extract information from images. Methods such as the Hough transform [29], [3] can be useful, but are not typically scale- or rotation-invariant. Selvage, Chenoweth and Gold [56] describe a modification to the chord transform, which was able to detect geometric structural content within an image. This transform has been shown to be scale-, translation- and rotation-invariant. Features in a series of aerial images were highlighted, showing an accurate extraction. The most noticeable problem with the chord transform, however, is its computational expense. Their variant chord transform

takes $O(N^4)$ computations where N is the number of edge pixels. This is impractical for real-time applications. They suggest a parallel SIMD (single instruction, multiple data) architecture to optimize the computation of the modified chord transform.

In recent years, there has been a trend towards using metrics derived from fractal geometry in the analysis of natural textures [42]. Pentland [49] used fractal-based descriptions of image texture to effectively characterize natural visual imagery. There have also been other works that attempt to apply fractal characteristics and measures to basic image analysis, such as texture segmentation [37], [48]. Using a fractional Brownian motion (fBm) model, Stewart et al. [60] demonstrated the application of fractal random process models as features in the analysis and segmentation of SAR imagery. The fractal dimension of natural textures, such as grass and trees, was computed and used as a texture feature in a Bayesian classifier.

Stewart, et al. discussed how metrics described by fractal geometry provide accurate measures of "roughness" and "irregularity" in scale-invariant natural forms. Moghaddam et al. [44], [45] have used the local fractal dimension, a fractal metric, for segmentation and analysis of infrared (IR) imagery. Likewise, Cooper [12] implemented a localized version of the fractal error measurement as well as other fractal metrics for the segmentation of aerial imagery. Stein [59] introduced another fractal error metric, calculated by a covering method similar to the method suggested by Peleg [48]. Solka, Rogers and Priebe [58] introduced a power law signature similar to Stein's. There were significant differences, however. Stein designed a discrimination scheme employing the slope and standard error of fit of the regression line, not unlike the algorithm the Cooper implemented [12], which is outlined in this work. Solka, Rogers and Priebe used different features to estimate fractal error. Also, Stein's decision rules were heuristic in nature, where Solka, Rogers and Priebe proposed advanced density estimation techniques in an effort to fully characterize the decision surfaces which originate from their power law signatures.

Image → | Edge Enhancement | → | Edge Linking | Edge Map →
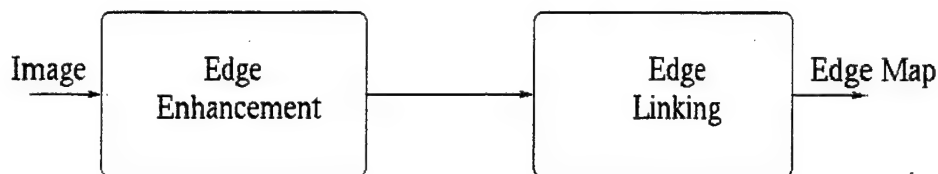
FIGURE 1 – The two-phase process for edge detection.

In another paper, Rogers, Solka and Priebe [53] outlined a method to calculate fractal dimension using a parallel distributed processing (PDP) approach. *A priori* boundary information was incorporated into their covering method, improving their segmentation results.

Cooper et al. [12] developed a fractal error metric based on the observed propensity of natural image features to fit a fractional Brownian motion model. They used this feature in a statistical classifier to successfully segment regions in aerial reconnaissance images. Jansing et al. [33] used this same feature to segment man-made objects in SAR imagery.

B.   Edge Detection

Edge detection has long been the subject of research in image processing and computer vision. Widely used in industrial vision systems, edge detection can be utilized for parts inspection or robot vision. Often, edges are detected and matched for registration in stereo vision. Edge detection is a two-phase process, as described by Figure 1.

The first phase is edge enhancement, which is the process of locating pixels in the image that may be considered step edges. This process typically involves the initial detection of edge points using such techniques as the Laplacian of the Gaussian [43], [5], [10]. Canny [9] did an extensive study on edge detection, deriving the optimal filter for edge detection. He discovered that the optimal filter was well-approximated

by the derivative of the Gaussian. The Prewitt edge operator [51] estimates the gradient with a quadratic fitting surface. Haralick [24] proposed an iterative facet model approach to edge enhancement. This model assumes that the spatial domain of the image can be partitioned into connected regions called facets. The boundaries of the facet define the step edges in the image. Jansing, et al. [32] proposed using a fractal error measure to highlight step edges.

Typically, false edges are eliminated from a set of criteria. For example, Canny [9] suggested thresholding edge points by comparing the magnitude of the gradient of an edge point with the magnitudes of its eight local neighbors. In Haralick's facet model [24], false edges are eliminated by a chi-squared test on the magnitude of the candidate edge pixel.

The second stage, edge linking, attempts to link edges which might be broken and discontinuous. This break in the edges may be caused by any number of phenomena in the image, including shadowing or noise. Some techniques for edge linking include the SEL algorithm [17] and the LINK algorithm [19]. Using search techniques, these methods attempt to produce edges that are connected over the image space.

CHAPTER III
AN INTRODUCTION TO FRACTAL ERROR

A. General Theory

1. Definition Of A Fractal

It is well known that many textures and scenes can be modeled as *fractals*. A fractal, according to La Brecque [6], "has a rough shape to one degree or another made of parts which, when magnified, resemble the whole." It is also well known that literature describing fractals often lacks precision when attempting to define what a fractal is. However, the reader of fractal geometry and theory can turn to Falconer [18] for a detailed description of the properties of fractals. Falconer states that

The set F is a fractal, if it has following properties:

1. F has a fine structure, that is, detail on arbitrarily small scales.

2. F is too irregular to be described in traditional geometrical language, both locally and globally.

3. Often F has some form of self-similarity, perhaps approximate or statistical.

4. Usually, the "fractal dimension" of F (defined in some way, and there are several unique definitions) is greater than its topological dimension.

5. In most cases of interest, F is defined in a very simple way, perhaps recursively (e.g., the Julia or Mandelbrot Sets).

2. Fractal Dimension

How are fractals distinguished between one another? How does one measure the size of fractal? What measure can be used to compare and contrast fractals?

Fractal dimension is the measure that is generally used to distinguish between fractals, giving the fractals a measurement of "size". This numeric representation attempts to quantify a subjective quality which one might have about how densely the fractal occupies the space in which it exists.

Fractal dimension is just as difficult to define as fractals themselves. Mandelbrot [42], Falconer [18], Peitgen et al. [47] and Edgar [16] provide excellent discussions of many different fractal dimension definitions. Each fractal dimension definition has a distinct style. Although the definitions are all related, Peitgen [47] claims that some definitions make sense in certain cases, while other definitions will not be cogent in the same case. Experience and heuristics prompt the selection of an appropriate fractal dimension definition, according to the application.

3. Lacunarity

Mandelbrot [42] defined another measure for fractals. *Lacunarity* describes the "holiness" ([36], pg. 236) of an occupied fractal lattice. The origin of the name lacunarity can be appreciated by looking at an image of cork in Figure 2. This image is part of a collection of texture images, presented by Brodatz [8]. From the Latin word "lacona," which means gap, lacunarity measures the gaps within a fractal structure. Thus, the percentage of spaces between the cork in Figure 2 is the measure of lacunarity. Practically, lakes or other natural objects within aerial images may be
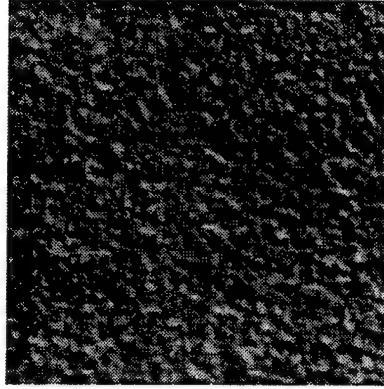
FIGURE 2 – Texture image of cork.

classified by using lacunarity as a feature measure.

4. Fractional Brownian Motion

Fractals may occur in many different forms. Mandelbrot [41] was the first to define *fractional Brownian motion* (fBm). Brownian motion is the model which refers to the erratic motion of small suspended particles, resulting from random collisions with other particles. Fractional Brownian motion is an extension of this model. Cooper [12] gives an excellent description of fBm; it will be summarized here.

The function of fBm is defined as the differences between successive samples. Let $B_H(\mathbf{t})$ represent a fBm signal, where $\mathbf{t}$ is a vector containing E independent variables. Then the increment of the fBm signal is described as $\Delta B_H = B_H(\mathbf{t_2}) - B_H(\mathbf{t_1})$, where $\mathbf{t_1}$ and $\mathbf{t_2}$ are two distinct measurements in time. The measure ΔB_H is normally distributed with a mean of zero and a variance proportional to the $2H$ power of the Euclidean distance. The mean takes the form of

$$\mathrm{E}[B_H(\mathbf{t_2}) - B_H(\mathbf{t_1})] = 0. \tag{1}$$

Likewise, the variance is defined as

$$\mathrm{Var}[B_H(\mathbf{t_2}) - B_H(\mathbf{t_1})] = k^2 |\mathbf{t_2} - \mathbf{t_1}|^{2H}. \tag{2}$$

where k^2 is the proportionality constant of the variance and H is the Hurst parameter, which must be $0 \leq H \leq 1$. Note that when $H = 0.5$, the fractional Brownian motion represents the original Brownian motion model. The value of H can be used to describe the fractal dimension D as

$$D = E + 1 - H, \tag{3}$$

where E is the Euclidean dimension (or, the number of independent variables of $\mathbf{t}$). It is therefore easy to see that small values of H produce high fractal dimension and large values of H produce a low fractal dimension. Combining the equations for mean and variance, while also taking into account the fractal dimension, we can arrive at the following relationship:

$$E[|B_H(\mathbf{t_2}) - B_H(\mathbf{t_1})|] = k|\mathbf{t_2} - \mathbf{t_1}|^H. \tag{4}$$

The above equation is the heart of the fractal error metric. The regions observed in an actual image may be a combination of many different textures. Each texture can be represented by its fractal dimension. But, before attempting to determine the fractal dimension, it is useful to know how well a region (or window) may fit the fractal model. Thus, measuring the error produced when estimating the fractal dimension will give a useful metric in order to determine the "fractalness" of a region in the image. A small error will indicate that a region fits the fractal model well, and thus can be considered a fractal. Conversely, a large error will indicate that the region fits poorly into the fractal model and thus is probably not a fractal and the fractal dimension measure is useless. Mathematically, this can be defined as

$$E[|G[\mathbf{x_2}] - G[\mathbf{x_1}]|] = k|\mathbf{x_2} - \mathbf{x_1}|^H \tag{5}$$

$$E[|\Delta G_{|\Delta \mathbf{x}|}|] = k|\Delta \mathbf{x}|^H \tag{6}$$

where G is the region or window in the image and $\mathbf{x}$ is the measured distances within

the region. Estimates of H and k can be found by using a linear regression scheme,

$$\ln \mathrm{E}[|\Delta G_{|\Delta \mathbf{x}|}|] = \ln k + H \ln |\Delta \mathbf{x}|. \tag{7}$$

These estimates, $\overline{H}$ and $\overline{k}$, can then be used to calculate the error with the following equation:

$$error_{|\Delta \mathbf{x}|} = \mathrm{E}[|\Delta G_{|\Delta \mathbf{x}|}|] - \overline{k}|\Delta \mathbf{x}|^{\overline{H}}. \tag{8}$$

Using a "center-oriented" window (i.e., a square window of n by n, where n is strictly odd), there will be five, nine, or fourteen error values, given the window is 5x5, 7x7, or 9x9 respectively. Thus, a cumulative error for the model can be given by the root mean square error

$$RMS\ Error = \sqrt{\frac{1}{n} \sum_{|\Delta \mathbf{x}|} (error_{|\Delta \mathbf{x}|})^2}. \tag{9}$$

Thus, using the RMS error, it is easily determined whether or not a pixel with a surrounding 5x5, 7x7 or 9x9 region is fractal in nature.

B. Fractal Error Algorithm

1. Description

Using the method described in the previous section, Cooper [12] developed an algorithm to calculate the fractal error for each pixel in a scene. This algorithm is described in detail in Table 1.

2. An Example

The following section outlines the steps of the entire fractal error algorithm. Note that this example will produce a single number that will represent the error for the center pixel in relation to its neighbors. Figure 3 represents a sample 5x5

TABLE 1

Fractal Error Algorithm

1. Define a 5x5, 7x7, or 9x9 sliding window.

2. Calculate each $\Delta\mathbf{x}$ and $E[|\Delta G|]$ for each pixel in the neighbor-hood of the sliding window.

3. Using linear regression, find the slope and the y-intercept for each unique set of $\Delta\mathbf{x}$ in the window from the equation $\ln(E[|\Delta G_{|\Delta\mathbf{x}_i|}|]) = \ln(k) + H|\Delta\mathbf{x}_i|$.

4. Assign $\overline{H}$ = slope and $\overline{k}$ = exp(y-intercept) from the above relationship.

5. Calculate the fractal error from $error_{|\Delta\mathbf{x}_i|} = E[|\Delta G_{|\Delta\mathbf{x}_i|}|] - \overline{k}|\Delta\mathbf{x}_i|^{\overline{H}}$, for that unique set of $\Delta\mathbf{x}$.

6. Calculate RMS error from $RMS \ Error = \sqrt{\frac{1}{n}\sum_{|\Delta\mathbf{x}_i|}(error_{|\Delta\mathbf{x}_i|})^2}$.

7. Save the RMS error for that pixel, move the window and repeat the process over the entire scene.

250	200	220	200	200
175	210	170	159	100
110	100	120	115	100
96	200	205	210	211
95	201	197	205	200

FIGURE 3 – A sample 5x5 window.

2.83	2.23	2.00	2.23	2.83
2.23	1.41	1.00	1.41	2.23
2.00	1.00		1.00	2.00
2.23	1.41	1.00	1.41	2.23
2.83	2.23	2.00	2.23	2.83

FIGURE 4 – The Euclidean distances over the 5x5 neighborhood.

window. Since the localized neighborhood is 5x5, there are five unique distances in the window, as shown in Figure 4.

The following table represents the absolute values of the differences of the gray levels over the unique sets of distances in the neighborhood. Thus, the gray scale value of each pixel that has a distance of 1.41 is subtracted from the gray scale value of the center pixel. Their absolute values are averaged to give $E[|\Delta G|]$ for each unique set of distances.

Using Equation 7, we can obtain estimates for the Hurst parameter and the proportionality constant, H and k, respectively. These estimates can be found using

TABLE 2

Distances From The Center Pixel (Δx) And The Expected Value Of The Absolute Values Of The Gray Scales In Reference To The Center Pixel ($E[|\Delta G|]$)

Δx	$\ln \Delta x$	$E[|\Delta G|]$	$\ln E[|\Delta G|]$
1.000	0.000	40.00	3.69
1.414	0.347	74.75	4.31
2.000	0.693	51.75	3.95
2.236	0.805	91.50	4.51
2.828	1.040	78.75	4.37

linear regression [2]. If the linear model takes the form

$$y = \beta_0 + \beta_1 x, \tag{10}$$

then estimates for β_1 (slope) and β_0 (y-intercept) are defined as

$$\widehat{\beta_1} = \frac{\Sigma(x_i - \overline{x})y_i}{\Sigma(x_i - \overline{x})^2} \tag{11}$$

$$\widehat{\beta_0} = \overline{y} - \widehat{\beta_1}\overline{x}, \tag{12}$$

where $\overline{x}$ is the sample mean of x and $\overline{y}$ is the sample mean of y. In our particular case, x represents the "$\ln \Delta x$" term in Equation 7 and y represents the "$\ln E[|\Delta G|]$" term in the same equation. To overcome the problem of evaluating $\ln E[|\Delta G|]$ when $E[|\Delta G|] = 0$, a protected natural log function, $\ln_p(x)$, was defined: If $x = 0$, then $\ln_p(x) = 0$, otherwise $\ln_p(x) = \ln(x)$. This protected log function prevents having to evaluate the natural log function at zero, whose value is minus infinity.

From that data in Table 2, it easy to see that $\overline{x} = 0.577$ and $\overline{y} = 4.17$. The estimates of β_0 and β_1 can thus be calculated,

$$\widehat{\beta_1} = \frac{\sum_{i=1}^{5}(x_i - 0.577)y_i}{\sum_{i=1}^{5}(x_i - 0.577)^2} = 0.585 \tag{13}$$

$$\widehat{\beta_0} = 4.17 - 0.577\widehat{\beta_1} = 3.83 \tag{14}$$
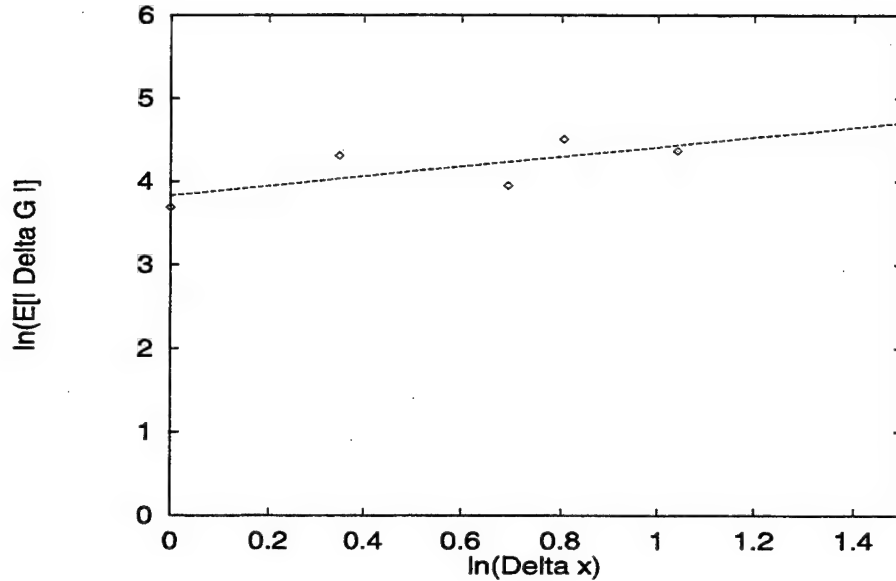
FIGURE 5 – $\ln \Delta x$ vs. $\ln \mathrm{E}[|\Delta G|]$ and the resulting estimated linear fit.

The Hurst parameter estimate, $\overline{H}$, is the slope of the linear model, that is, $\overline{H} = \widehat{\beta_1}$. The proportionality constant estimate, $\overline{k}$ is the y-intercept of the linear model, $\overline{k} = \exp(\widehat{\beta_0})$. Therefore, $\overline{H} = 0.585$ and $\overline{k} = 46.1$. Figure 5 shows $\ln \Delta x$ vs. $\ln \mathrm{E}[|\Delta G|]$ and the estimated linear fit.

Errors with respect to each unique distance set can be calculated with Equation 8. Table 3 shows the result of using Equation 8 in this example.

The overall RMS error, as defined by Equation 9, is 14.31. This number represents the "fractalness" of the center pixel in relationship to its neighbors. Decisions regarding the fractalness of the pixel are typically made in reference to the entire image. Thus, if the variation of fractal errors is from 0 to 15, this pixel is not likely to be fractal in nature.

3. Results

In this section, the algorithm is applied to several images. The original image and the resulting fractal image are shown.

TABLE 3

Calculated Error From Fractional Brownian Motion Model

Δx	error
1.00	-6.06
1.41	18.4
2.00	-17.3
2.24	17.7
2.83	-5.9

Cultural objects in the aerial images, Figures 6 and 7, are highlighted. The pixels representing cultural objects are bright in comparison to the other pixels in the image, indicating that fractal error is high. In these examples, fractal error proves to be a useful discriminating feature for segmenting cultural objects in aerial images.

Fractal error will also enhance the edges in an image, as shown in Figures 8 and 9. Since objects that are less fractal in nature have higher fractal error values than more naturally occurring objects, the normalized fractal error can be viewed as a likelihood measure. This likelihood will provide a useful metric in enhancing and linking edges in an image.

Computational expense is the unfortunate by-product of this useful algorithm. Many applications require real-time computation to be practical. The next chapter will introduce a genetic approach to approximating fractal error. This genetic approximation to fractal error provides real-time analysis, whether for cultural object detection or edge detection.
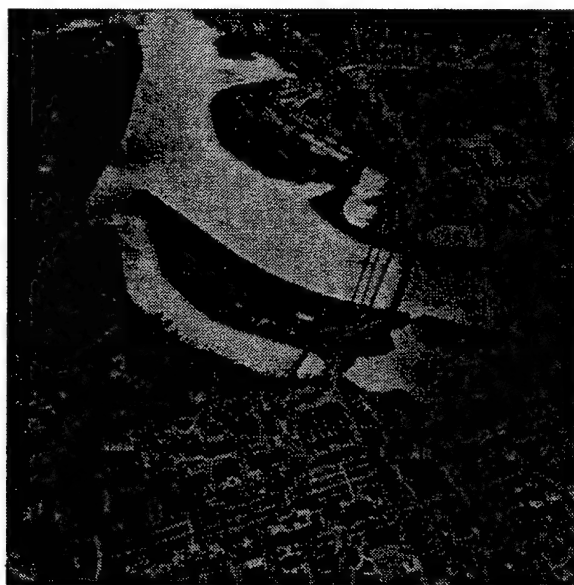
(a) Original Image
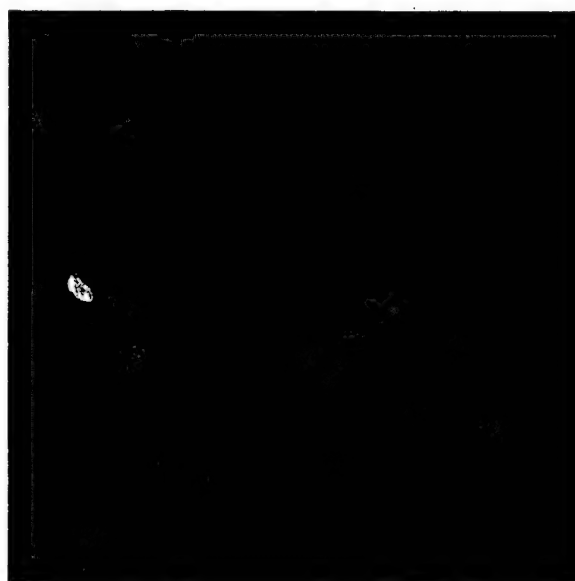
(b) Fractal Error Image

FIGURE 6 – Washington, DC image.

(a) Original Image
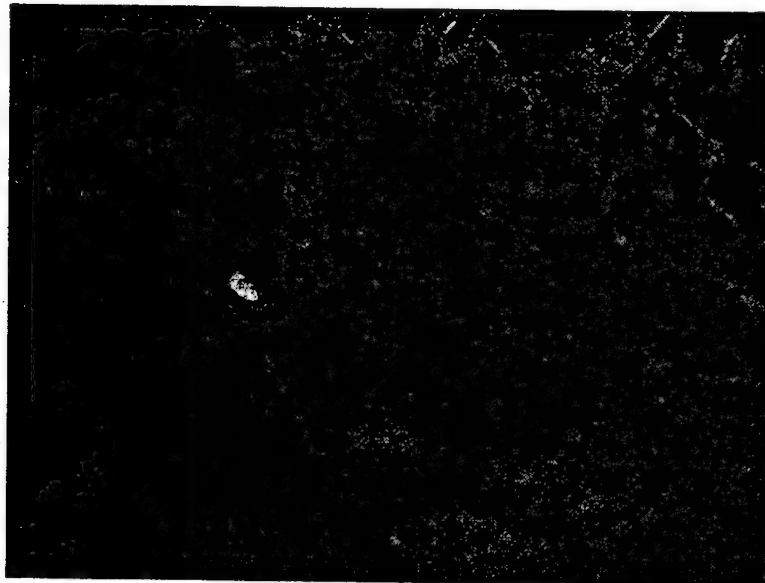
(b) Fractal Error Image

FIGURE 7 – SAR image of desert terrain.
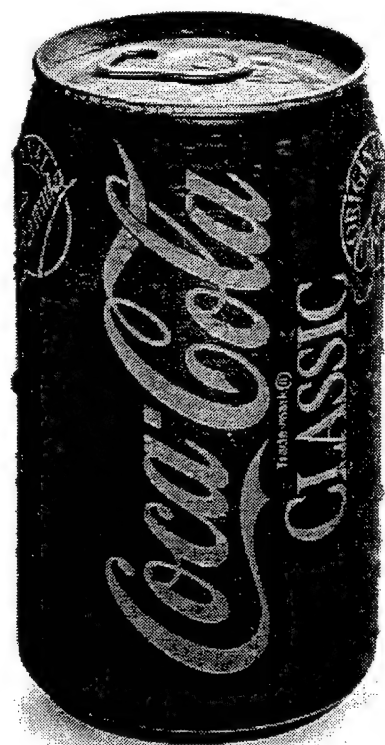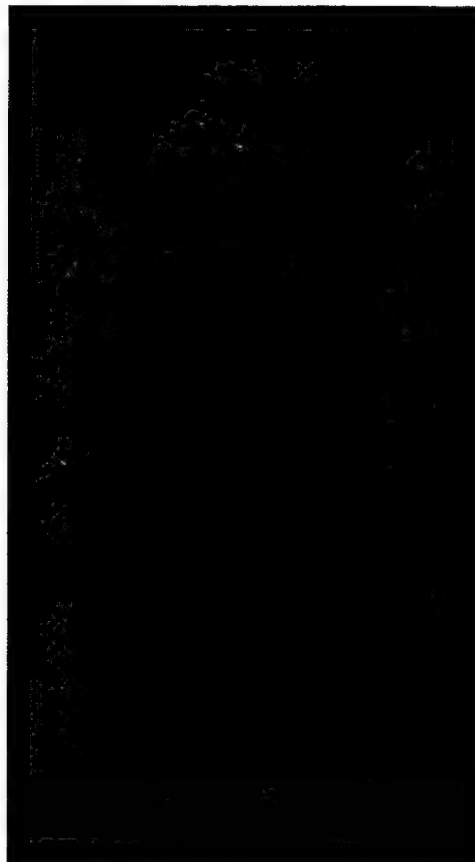
(a) Original Image   (b) Fractal Error Image

FIGURE 8 – Coke can image.
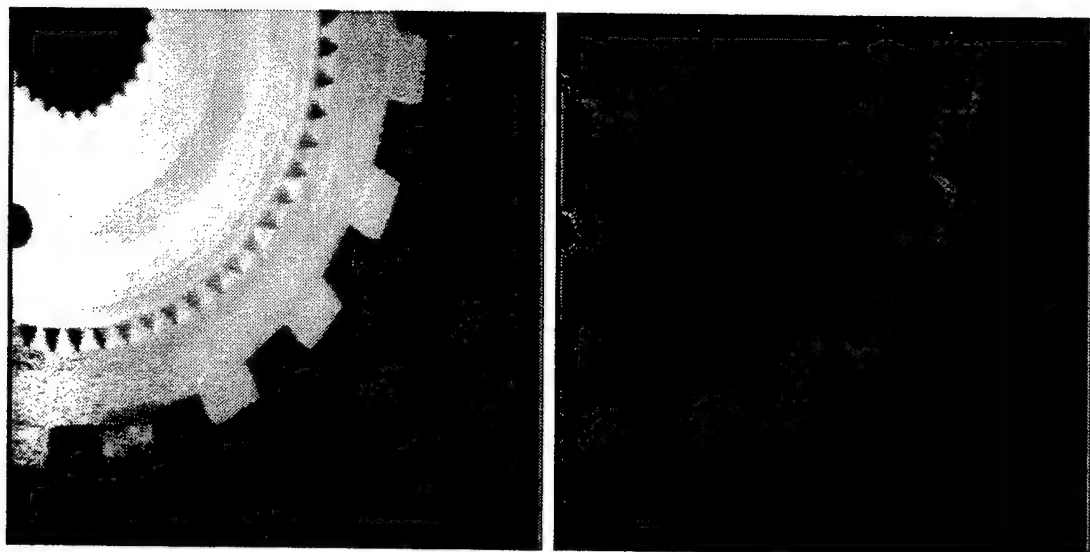
(a) Original Image   (b) Fractal Error Image

FIGURE 9 – Gear image.

CHAPTER IV
GENETIC APPROXIMATION TO FRACTAL ERROR

A. Introduction

In 1975, John Holland [28] introduced the publication *Adaptation in Natural and Artificial Systems.* In this publication, Holland integrated two themes that had persistently appeared in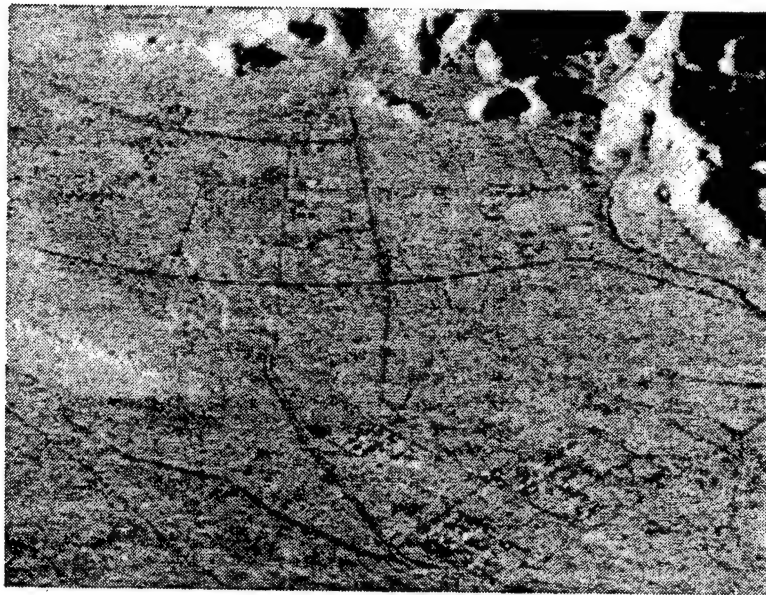 his studies: the representation of complex structures using simple representations (such as bit strings), and the ability to modify and improve such structures with simple transformations. Holland demonstrated that under the proper conditions, bit strings could "evolve" into improved bit strings. The improvement was dependent on the fitness of each member of the group of strings. Holland realized the tremendous potential of such an operation, and thus, initiated the study of *genetic algorithms.*

Holland discovered that the fundamental structures of genetics, chromosomes, can also be analogous to structures in other fields, such as types of goods in economic planning, strategies in gaming theory and functions in artificial intelligence. If the structures could be represented in a similar fashion, Holland posed the question of whether the same sort of operators that evolve chromosomes could be used to evolve optimal structures in the other fields? Holland demonstrated that, in fact, they do. Just as the operations of mutation and recombination adapt chromosomes to function in specific (and hopefully optimal) ways, the operations of production activities in economic planning or learning rules in artificial intelligence can adapt their respective structures.

In the same way chromosomes evolve over several generations, so do the structures in our genetic algorithm. Structures are allowed to sexually reproduce into the generation via some random operation according to the structure's fitness in the population. Typical fitness functions may include utility in economic planning, error functions in controls, payoff in gaming theory, or comparative efficiency in artificial intelligence. Those individuals who are more fit than others (the lowest error in the case of controls or the highest payoff in the case of gaming theory) have a higher probability of reproducing into the next generation. As the total population evolves over generations, the best parts of fit structures combine with the best parts of other fit structures, creating structures that generate better fitness values. Holland [28] demonstrated that even in large, complicated search spaces, given certain conditions on the problem domain, the genetic algorithm would tend toward the global extrema.

Much has been written about genetic algorithms and it would be impossible to cite completely in this work. The reader should reference Holland [28], Goldberg [21], Davis [14], or Koza [39] for more detailed discussion of the theory behind genetic algorithms. While Holland, Goldberg and Davis primarily focused on genetic algorithms, Koza has spent much of his studies extending genetic algorithms to *genetic programs* (GP). Here the structures were actual computer code, typically in LISP or C. These structures of code were then allowed to evolve over hundreds or thousands of generations, until the needed problem-specific code has been found. Using SPICE (Simulation Program with Integrated Circuit Emphasis), Bennett et al. [4] developed a GP to design a low-distortion, low-bias 60 dB (1000-to-1) amplifier with good frequency generalization. They contended that the performance of a GP can match or exceed human performance in some circuit design problems [40]. In 1996, Harris and Buxton [25] demonstrated that GP techniques can be used to evolve 1-D edge detectors.

Because fractal error can be computationally expensive, particularly for large

images, an approximation for real-time processing is desirable. This chapter will explore an approximation to fractal error with a genetic algorithm. An outline of the simple genetic algorithm (SGA) will be introduced to lay the foundation for the genetic approximation to fractal error (GAFE). Included in the discussion of the SGA, an example involving genetic algorithms will also be examined.

B.   The Simple Genetic Algorithm

1.   General Theory

The genetic algorithm evolves a set (the population) of individual structures (the chromosomes) into a new population using operations modeled after the Darwinian principle of reproduction and survival of the fittest. The structures are typically fixed-length character strings, often representing binary numbers. Each string in the new population, also known as the next generation, is evaluated for fitness. If the solution has not yet been reached, the process is repeated and a new population is born from the structures of the previous generation.

The simple genetic algorithm will be described in this section with an example of finding the global maximum of the monotonically increasing function,

$$f(x) = x^2 \text{ where } 0 \leq x \leq 15. \tag{15}$$

Because the function is monotonically increasing, it is not difficult to see that the global maximum lies at $x = 15$. However, we will suspend this knowledge in order demonstrate how the SGA will converge to the solution.

The basic operation of the genetic algorithm is described by the flowchart in Figure 10. At the start of the algorithm, a random population called the 0th or initial generation is created. Each member of the initial population is evaluated for fitness, while checking to see if the desired solution has been reached. If no solution is found
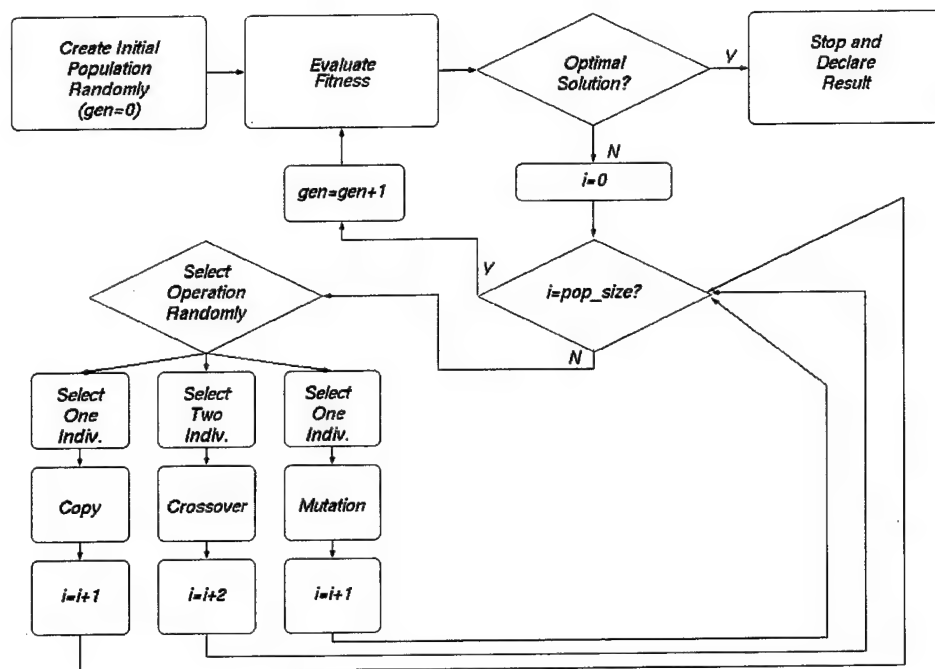
FIGURE 10 – Flowchart of the simple genetic algorithm.

in the initial population, which is highly likely, then the next generation is formed. The formation of the next generation is performed by probabilistically selecting a genetic operation, selecting one or two fit individuals, performing the selected operation and inserting the new individual or individuals in the new population. The new population is then evaluated for individual fitness. If a solution is found in this new population, the algorithm is stopped and the result is reported. However, if no solution is obtained, the process repeats again until a solution is found or a predetermined number of generations have been created and evaluated.

In traditional min-max theory, the global extrema are determined by evaluating the first derivative at zero. In some deterministic search methods, statistical knowledge of the function must be known or assumed [31]. In blind search methods, such as Monte Carlo methods [22], nothing regarding the function need be known, but there is no guarantee that the global extrema will be found in a timely fashion. Genetic algorithms, however, do not operate on the function directly, but on a pa-

FIGURE 11 – A 4-bit integer representation for the search space $[0, 15]$.

rameterization of the inputs to the function. In this manner, no *a priori* information regarding the function need be known. The genetic algorithm is also not limited to just global optimization, but to a number of problems, such as gaming theory, scheduling, and economic planning.

There are several problems associated with setting up this algorithm. How does one determine the structure of each individual in the population? How is the fitness function defined, and is it problem-dependent? How is an individual chosen out of the population, that is, how does one implement "survival of the fittest?" Are there rules or guidelines in randomly selecting the genetic operation? How many individuals should be in the population? How many generations is enough? Before the genetic algorithm can be used, each one of these issues must be addressed.

a. Parameterization. Holland and many of his students [14] structured their chromosomes in bit strings. Many parameterizations can be set up as bit representations. In the case of our example, since it is known that the search will be conducted in the interval $[0, 15]$, the bit representation could be a 4-bit binary number, as shown in Figure 11. This configuration would only represent the integers within the search space. If a real number representation is required, then a number of bits could be used to encode the fractional part of the number, such as the example in Figure 12. We will use the 4-bit integer representation for our SGA.

There are certainly other ways to represent the chromosome structure. Examples of other representations may include ordered lists (for bin-packing or the traveling

FIGURE 12 – A 6-bit fixed-point representation for the search space $[0, 15]$.

salesperson problem), embedded lists (for scheduling), or variable- element lists (for semiconductor layout). Each problem will lend itself to a particular structure.

b. *Fitness Functions.* Finding a proper fitness function is critical to the development of a genetic algorithm. The fitness function is the deciding factor in which an individual will be copied, mutated, or crossed over with another individual in the formation of a new population. Here, the value of $f(x)$ may be used as the fitness measure. A higher value of $f(x)$ represents a more fit individual. In the case of curve fitting, RMS error can serve as the fitness function, where smaller RMS errors denote fitter individuals.

Normalization plays an important role in assigning fitness values. When the raw numbers of a fitness function are used, a very good individual may score 110 and a poor one may score 100. If these raw values are used without alteration in the reproduction of the next population, it will take some time before the children of the good individual outnumber those of the poor one. For this reason, it may be desirable to normalize the scores in some fashion. However, if the performance of the genetic algorithm is too sensitive to the normalization method chosen, it will stress the improved individuals too much. This will lead to inbreeding with the population, promoting the rapid dominance of a single strain of structures. In this case, it is likely that the genetic algorithm will converge to a local maxima or minima and not the global solution. This is called *premature convergence.*

c. Selection. There are several methods of selecting fit individuals in the population for reproduction. The most common form of selection is the "roulette wheel." This method models a roulette wheel in that there are a fixed number of slots on the wheel. Slots are allocated according to fitness, with the highest fit individuals getting the most number of slots and the poorest fit individuals getting the fewest slots or perhaps no slot at all. Following a uniform distribution, the wheel is spun each time to select individuals for reproduction. There is no guarantee that the most fit will be chosen, but there is a high probability that a structure having a good fitness measure will be chosen. Unfortunately, this method often favors the most fit chromosomes frequently and the genetic algorithm suffers from premature convergence.

Tournament selection is another popular method of selection, whereby Darwin's idea of "survival of the fittest" is modeled. Two structures in the population are randomly chosen. The individual with the best fitness value is then selected for reproduction. This method avoids the problem of premature convergence, keeping the population as diverse as possible while still advancing the population towards the desired goal.

d. Crossover. Once an individual (or two individuals in the case of crossover) is selected. reproduction into the next generation may begin. Crossover is a reproduction operator that combines properties of two parent chromosomes into two children chromosomes. Figure 13 shows simple single-point crossover. If the individuals are of length n, then a point between $[1, n]$ is chosen randomly from a uniform distribution. The children are formed by combining the genetic material from the parents. The newly formed offspring are then introduced to the new population. The parents are not deleted or destroyed from the old population, giving them the opportunity to reproduce in the future.

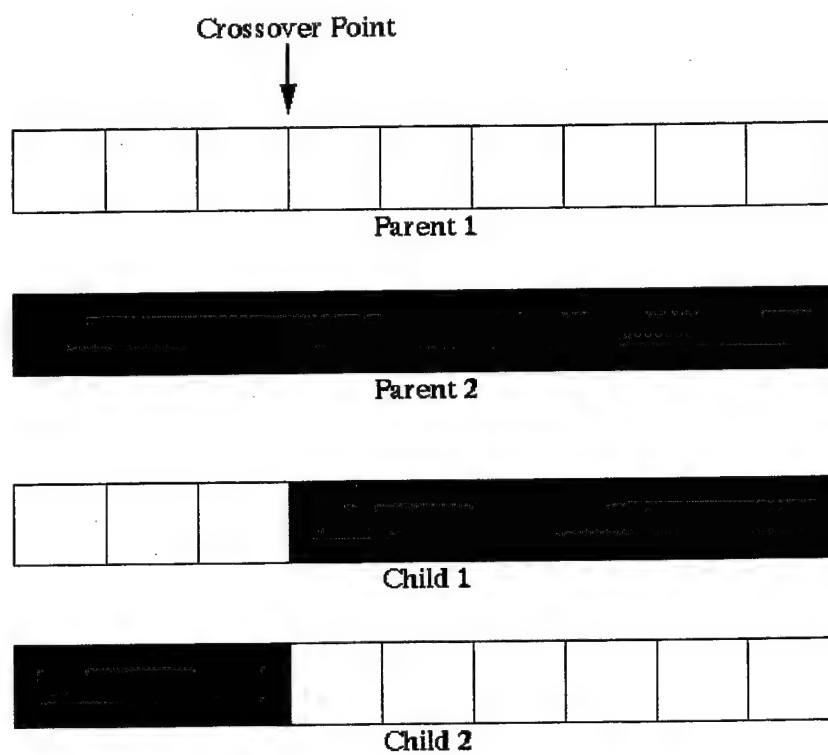Crossover is not restricted to single-point crossing. Multi-point crossover has

FIGURE 13 – Illustration of single-point crossover.

also been shown to be useful in reproduction from one generation to the next. Single-point crossover tends to limit the areas of search, thereby risking premature convergence. Multi-point crossover, which uses two or more crossover points, creates diverse children, expanding the search area, and thus increasing the probability of finding the optimal solution. Crossover is the central operation of reproduction in genetic algorithms. Thus, the probability of choosing crossover, called p_c, is usually 0.9.

e. *Mutation And Copy.* In addition to crossover, there are two other reproduction operators: mutation and copy. Mutation is the act of changing one or more bits in a parent string. The mutated string then gets passed as a child to the next generation. Figure 14 illustrates single-point mutation. A bit within the genetic material is chosen randomly from a uniform distribution. If the bit is a 0, it mutates to a 1. Conversely, if the bit is 1, it changes to 0. Holland [28] described the mutation operator as a "background" operator. It maintains a full range of individuals for the crossover operator; this prevents the adaptive nature of reproduction from being trapped on local optima. Typically, the probability for choosing the mutation operator (p_m) is set low, perhaps at 0.1 or lower. Too little mutation produces premature convergence and inbreeding; too much mutation causes the population to oscillate between search spaces, hindering convergence.

The copy operator simply allows a parent from the previous generation to pass to the next generation. This increases the probability that strings with high fitness values will enter into the next generation and maintain the current search space. This is particularly useful if the population resulting from crossover and mutation is weak. The probability of choosing copy is defined as $(1.0 - (p_m + p_c))$.

f. *Migration.* Introduced by Potts et al. [50] in 1994, the migration operator helps combat the problem of premature convergence. Potts et al. allowed multiple sets of individuals to evolve using traditional genetic operators such as crossover and

Mutation Point

Parent

Child
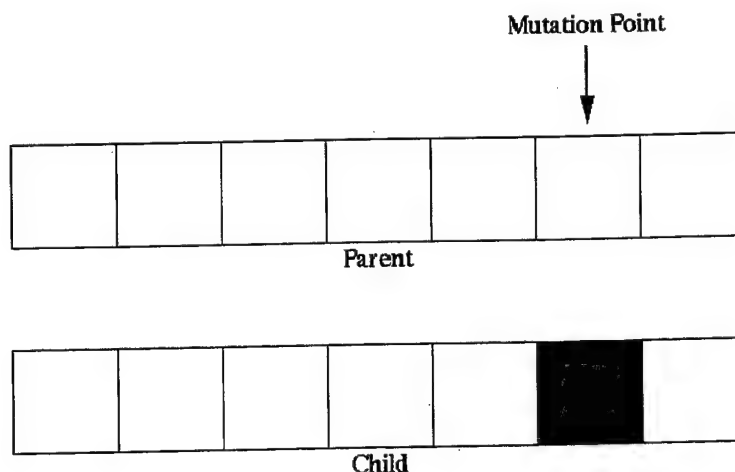
FIGURE 14 – Illustration of the single-point mutation operator.

mutation. Each set of individuals acts as an independent population, like a tribe or village. The migration operator chooses individuals at random and moves them to another population. This interaction prevents inbreeding and promotes reproduction among those individuals that have good characteristics, that is, high fitness values.

The migration operator used in this research is an extension of the method proposed by Potts et al. [50]. Migration is implemented here by randomly generating a new individual and placing it in the new population. This randomness may allow for movement to a search space that the individuals in the previous population were unable to reach. This modification allows for the property of migration without the overhead of multiple populations.

Premature convergence can be overcome by the choice of reproduction operators and population size, as well as variations on the operators. Adjustments of parameters such as population size, crossover rate, and mutation rate may also contribute to or hinder premature convergence. There are no generalizations to choosing a combination of operators and parameters to ensure convergence on the desired solution. Thus, the area of optimal genetic algorithm design is a prominent area for further research.

C. How The Genetic Algorithm Works

It is not easily seen how this adaptive system works. However, consider the following: an SGA is attempting to optimize the monotonically increasing signal in Equation 15. The input x has been encoded in a 4-bit binary string, representing the search space $[0, 15]$. The population size has been fixed at 4 individuals per generation. In the initial population, the strings 0001, 1101, 0010, and 0011 were created. Each string has a fitness of 1, 169, 4, and 9, respectively. Having the highest fitness, gene 1101 is the best candidate for reproduction.

The possible search space that each string, or set of strings, represents can be described by schemata. Schemata is defined by the set $0, 1, *$ where $*$ is a "don't care" condition. Each schemata describes an area that can be searched, while each gene represents a point in that area. For example, consider the schema represented by 00**. This represents the search interval $[0, 3]$. The third and fourth chromosomes (0010 and 0011) in the above example reside in this search space. Offspring from either of these two individuals are likely to fall within this area. However, since the fitnesses of the third and fourth chromosomes are relatively low, it is unlikely that they will reproduce.

This schema gives some indication as to why 0010 and 0011 do so poorly in the population. The chromosomes, and indeed the schema 00** itself, represent the farthest possible area from the maximum in the search space. In fact, the fitness for 00**, which can be represented as an average value, would still be much less than the fitness for 10**, which is much closer to the maximum. Thus, any schema describes a set of points from the search space of a problem that have certain specified similarities [39]. The selection and reproduction operators exploit the similarities, creating a new population with offspring that exhibit the schema of their parents. This is seen in nature when an individual survives to the age of procreation and actively reproduces, his or her offspring carry their "best" characteristics, such as eye

However, if the a higher value of the raw fitness is better, a standard fitness can be found by

$$s(i, t) = r_{max} - r(i, t), \tag{18}$$

where r_{max} is the maximum possible value of raw fitness.

In addition to normalized and standardized fitness measures, Koza [39] described a third modification to raw fitness values called adjusted fitness. The adjusted fitness values are calculated from the standardized fitness values,

$$a(i, t) = \frac{1}{1 + s(i, t)}. \tag{19}$$

The adjusted fitness, like the normalized fitness, lies between 0 and 1. Bigger values of $a(i, t)$ indicate better individuals. The benefit of adjusted fitness values is that adjusted fitness has a tendency to exaggerate the importance of small differences in the value of $s(i, t)$, as $s(i, t)$ approaches zero. It should be noted that standardized, normalized and adjusted fitness are not relevant for tournament selection.

In the example, it is highly likely that 1101 will be chosen for reproduction, provided the opportunity is available to produce offspring from other strings (1101 holds 92% of the slots). This example shows the utility of tournament selection. Tournament selection may provide a more diverse population in the next generation. It is highly likely that roulette wheel selection will provide an inbred population, most likely ending up in premature convergence at the point $x = 13.0$. Population size also contributes to this problem. If the population size is too small, the fitness values may converge upon a local minimum or maximum, ending up with a population of clones. On the other hand, if the population is too large, it will take a considerable time to locate the schema necessary to converge on the perfect solution. Populations that are too small die quickly and populations that are too big oscillate back and forth between search spaces.

D. Genetic Approximation Of Fractal Error (GAFE)

To generate an approximation of fractal error, a mathematical structure is needed. Much of the information in calculating fractal error resides in the expected value of the absolute differences of the gray levels from the center pixel, grouped in uniform distances from the center pixel (refer to Chapter III). This is expressed mathematically as $E[|\Delta G_{|\Delta x_i|}|]$. Weights are assigned to each of these expected values and all of the weighted expected values are added together to form the approximation of fractal error. In the case of the 5x5 window, this is expressed as

$$
\begin{aligned}
\widehat{FE} \ = \ & a_0 \ln(E[|\Delta G_{\Delta x=2.82}|]) + a_1 \ln(E[|\Delta G_{\Delta x=2.23}|]) \\
+ \ & a_2 \ln(E[|\Delta G_{\Delta x=2.00}|]) + a_3 \ln(E[|\Delta G_{\Delta x=1.41}|]) \\
+ \ & a_4 \ln(E[|\Delta G_{\Delta x=1.00}|]).
\end{aligned}
\tag{20}
$$

The genetic approximation of fractal error, named GAFE, evolves the weights a_0, $a_1, \ldots, a_4$ to try to estimate fractal error. The natural log of the expected values is used to maintain the linearity of the measure. The probability for crossover is set at 0.7. and the probability for mutation, copy, and migration are set at 0.1. Each operator is chosen randomly from a uniform distribution. The population size and the maximum number of generations allowed for each simulation is variable. Fitness for each individual is determined from signal-to-noise ratio (SNR). SNR measures the ratio of original signal to noise in a given test image. It is defined as

$$
SNR = \frac{\sum_{r=0}^{M} \sum_{c=0}^{N} I_m(r,c)^2}{\sum_{r=0}^{M} \sum_{c=0}^{N} (I_m(r,c) - I_o(r,c))^2}
\tag{21}
$$

where $I_m(r,c)$ is the measured image and $I_o(r,c)$ is the original signal for MxN sized images. The lower the value of SNR, the noisier the signal is. Thus, to evolve weights with a genetic algorithm, it is desirable to maximize the signal-to-noise ratio.

The GAFE is also written in parallel to ensure fast simulation times. The evaluation of fitness for each individual in the population consumes most of the run

TABLE 5

Several Trials Of The GAFE With Varying Population Size And Generations

Trial	Population Size	Maximum Generation Size	SNR
A	100	10000	1.27
B	500	100	1.52
C	500	2000	1.56
D	1000	2000	1.63
E	2000	250	1.74
F	10000	90	3.82

time for the GA itself. To parallelize the algorithm, each individual is evaluated for fitness separately, running on as many CPUs as are available. The parallel algorithm was executed on an SGI Onyx R10000 high-performance computer, with 16 CPUs.

Table 5 shows six different trials of GAFE and their results. The varying population size and maximum number of generations allowed are also given. The training image for each of these simulations is shown in Figure 15.

Figure 16 illustrates the resulting image from each trial. Figures 17 through 22 show the change in SNR over the generations of the genetic simulations.

Each individual is comprised of 100 bits, 20 bits per weight. The first 19 bits are the magnitude, starting at 2^2 and ending at 2^{-16}. The 20th bit for each weight is a sign bit, allowing for positive and negative weights. Thus, the interval for each 20 bits is $(-8, 8)$. Table 6 shows the final weights for Trial F.

The results are revealing. For very small populations, sized from 100 to 1000 individuals, the GAFE is unable to isolate a search area that will find a suitable set of weights, that is, it incorrectly converges to a local extrema. It is clear that larger populations fare better than smaller populations, regardless of the number of

(a) Original Image
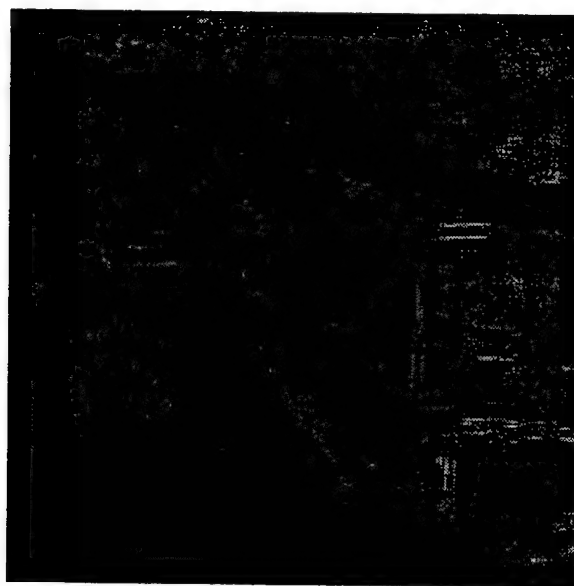
(b) Actual Fractal Error Image

FIGURE 15 – Training image for the genetic approximation to fractal error (GAFE).

TABLE 6

Final Weights From GAFE, Trial F

$$a_0 = 0.0022583$$
$$a_1 = 0.0969543$$
$$a_2 = -0.0341644$$
$$a_3 = -0.00694275$$
$$a_4 = -0.000930786$$

generations evolved. Even in the final experiment, where there were 10,000 individuals in the pool, the final solution was found by chance. It is unclear whether the optimal solution would be converged upon, even if the gene pool were allowed to evolve for many generations. This may be attributed to a number of differing factors, such as the actual mathematical structure of $\widehat{FE}$, the parameterization of the weights, or the choice of genetic operators and their probabilities. However, it is not difficult to see that the weights evolved in Trial F produce a usable, if not noisy, image.

Figures 23-26 show the results of the GAFE for other images, using the weights found in Trial F. Table 7 outlines the SNR between the original fractal error images and their resulting approximations. Table 8 compares computation times for the fractal error algorithm and the GAFE. There is approximately a 30 % decrease in computation time with the GAFE.

The evolved expression appears to function for this limited set of images, providing the needed features (either edges and/or cultural objects) in the presence of some noise. The noise in the images does not hinder general edge detection or segmentation. Simple thresholding can be employed to filter out the noise, because its gray-scale values are much higher than that of the desired features in the GAFE images.

TABLE 7

SNR For Figures 23-25

Image	SNR
Washington, DC Image	2.41
SAR Image of Desert Terrain	5.08
Gear Image	1.57
Coke Can Image	2.66

TABLE 8

Computation Time For The Original FE Algorithm and The GAFE

Size	FE Algorithm (seconds)	GAFE (seconds)
256x256	1.43	0.43
227x425	2.14	0.64
589x409	5.59	1.65
512x480	5.68	1.69
974x723	16.36	4.85
1267x941	27.74	8.22

The mathematical structure of GAFE poses a unique problem. There is no generalized form of fractal error as it is defined in Chapter III. A generalized form would make it possible to generate a type of convolution mask which could quickly and easily be used to compute fractal error. Forcing a mathematical structure to find the approximation is restricting. In the future, another approach may evolve a mathematical expression using a GP. This would expand the search space from merely searching the weights for $\widehat{FE}$ to searching the infinite space of reusable functions to find the approximation.
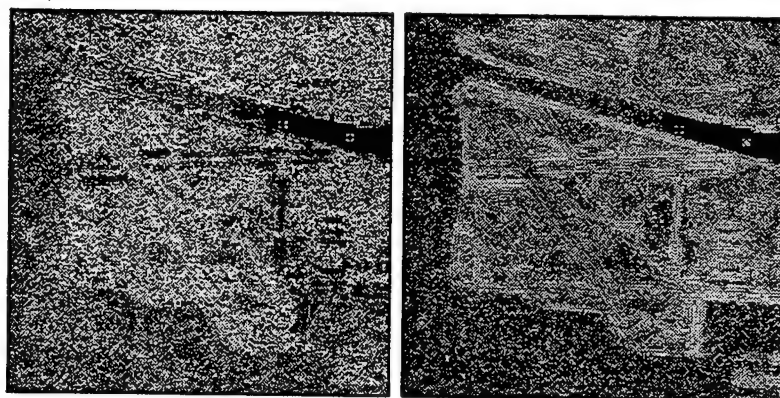
(a) Trial A

(b) Trial B
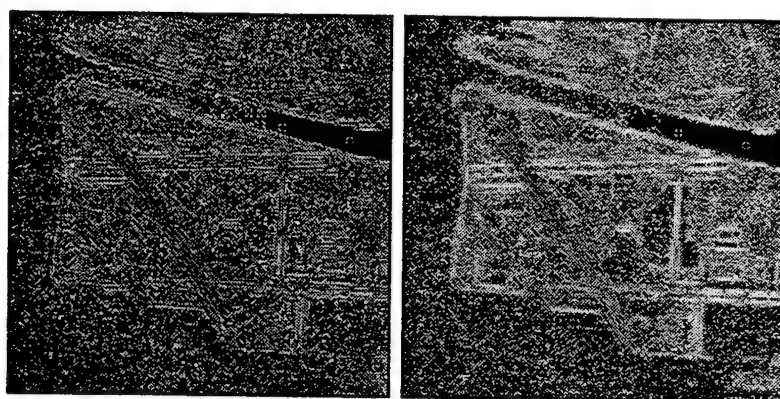
(c) Trial C
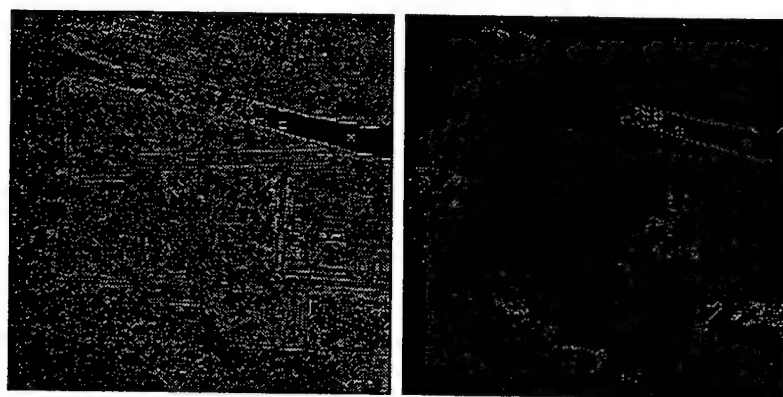
(d) Trial D

(e) Trial E

(f) Trial F

FIGURE 16 – Resulting $\widehat{FE}$ images from each trial described in Table 5.
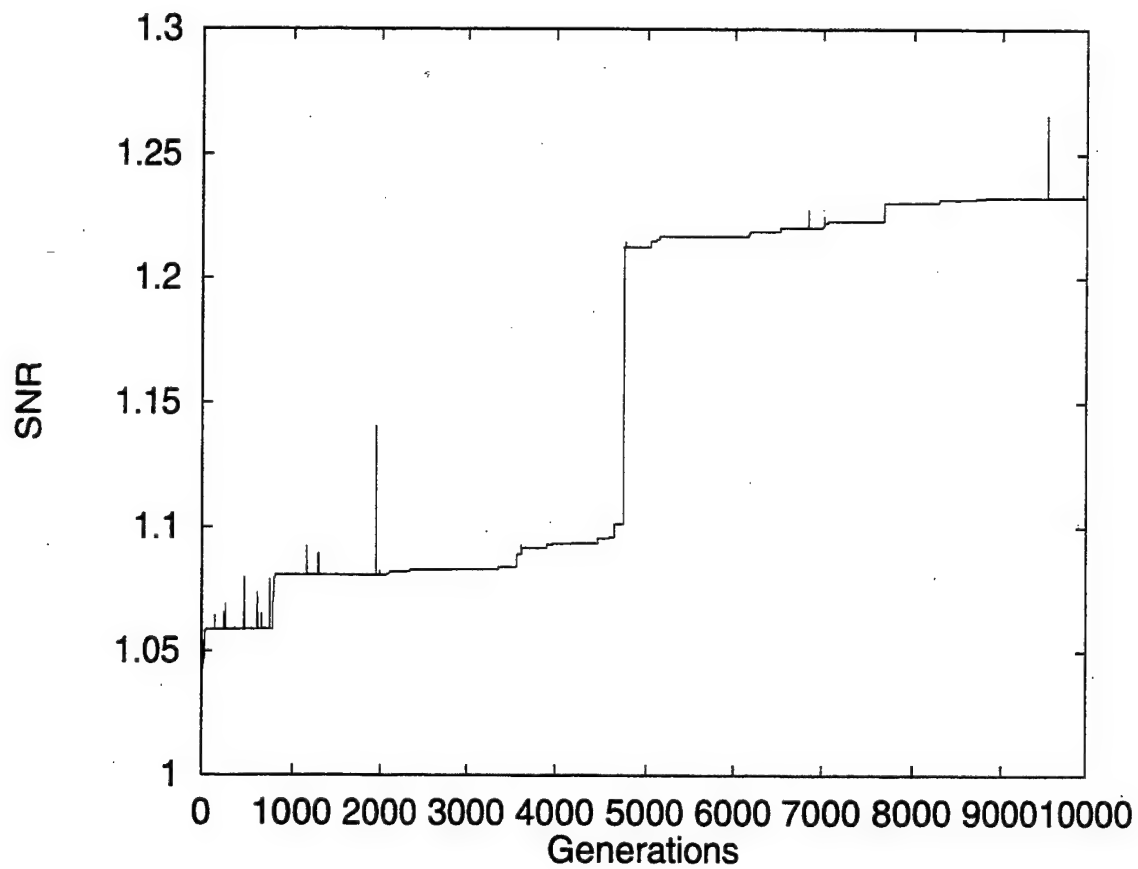
FIGURE 17–SNR vs. number of generations for Trial A.
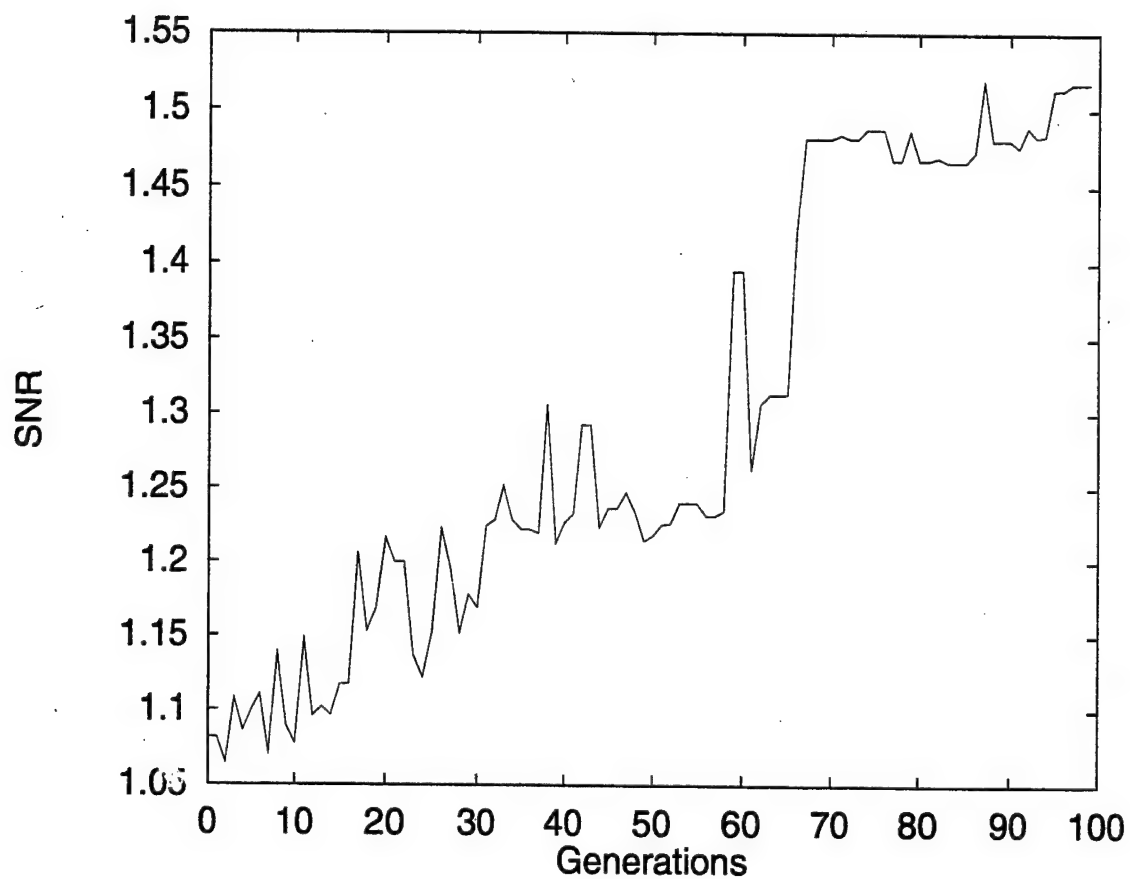
FIGURE 18 – SNR vs. number of generations for Trial B.
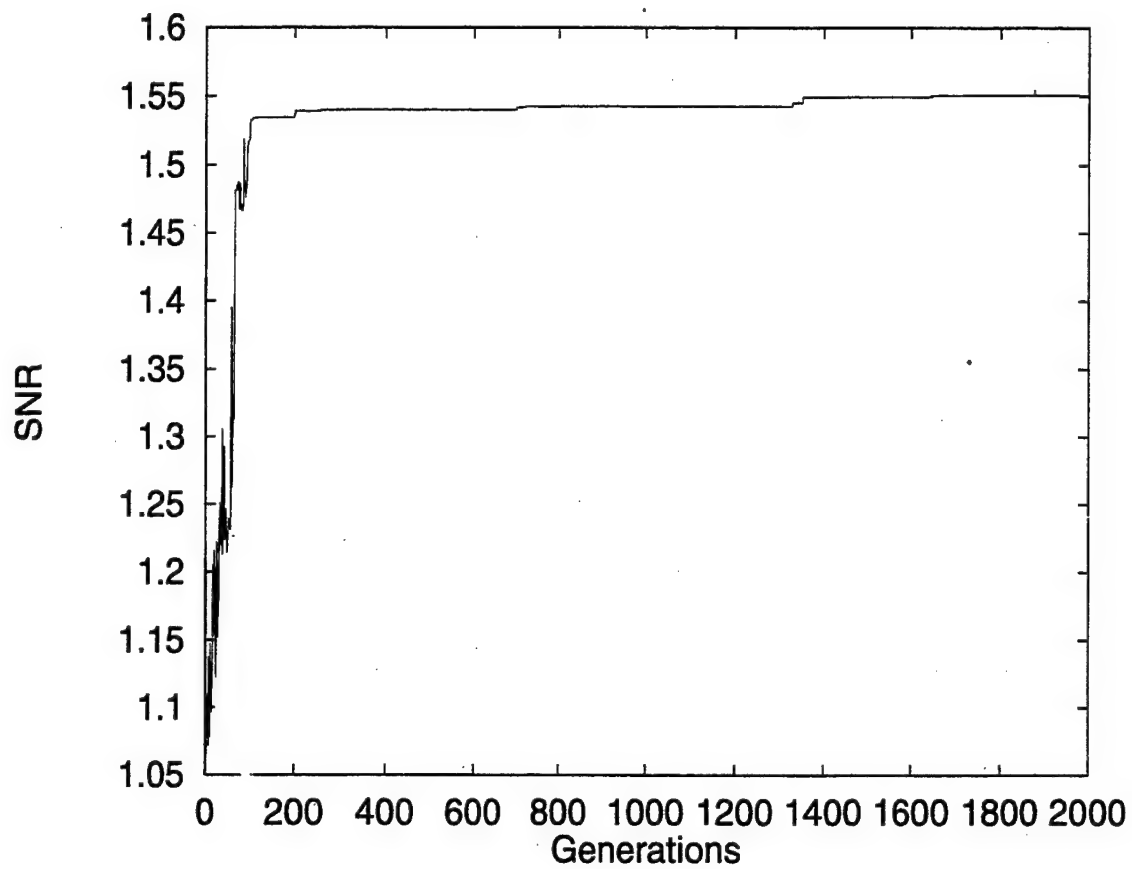
FIGURE 19 – SNR vs. number of generations for Trial C.
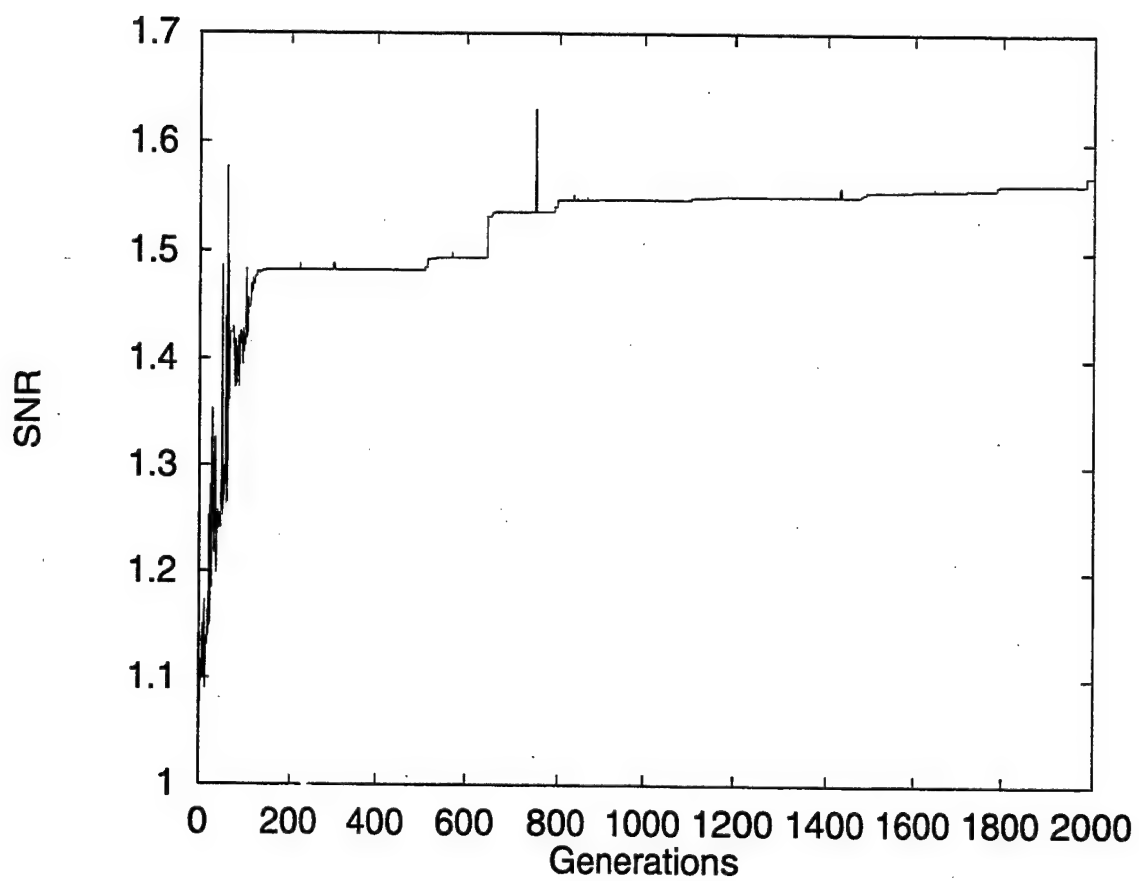
FIGURE 20 – SNR vs. number of generations for Trial D.
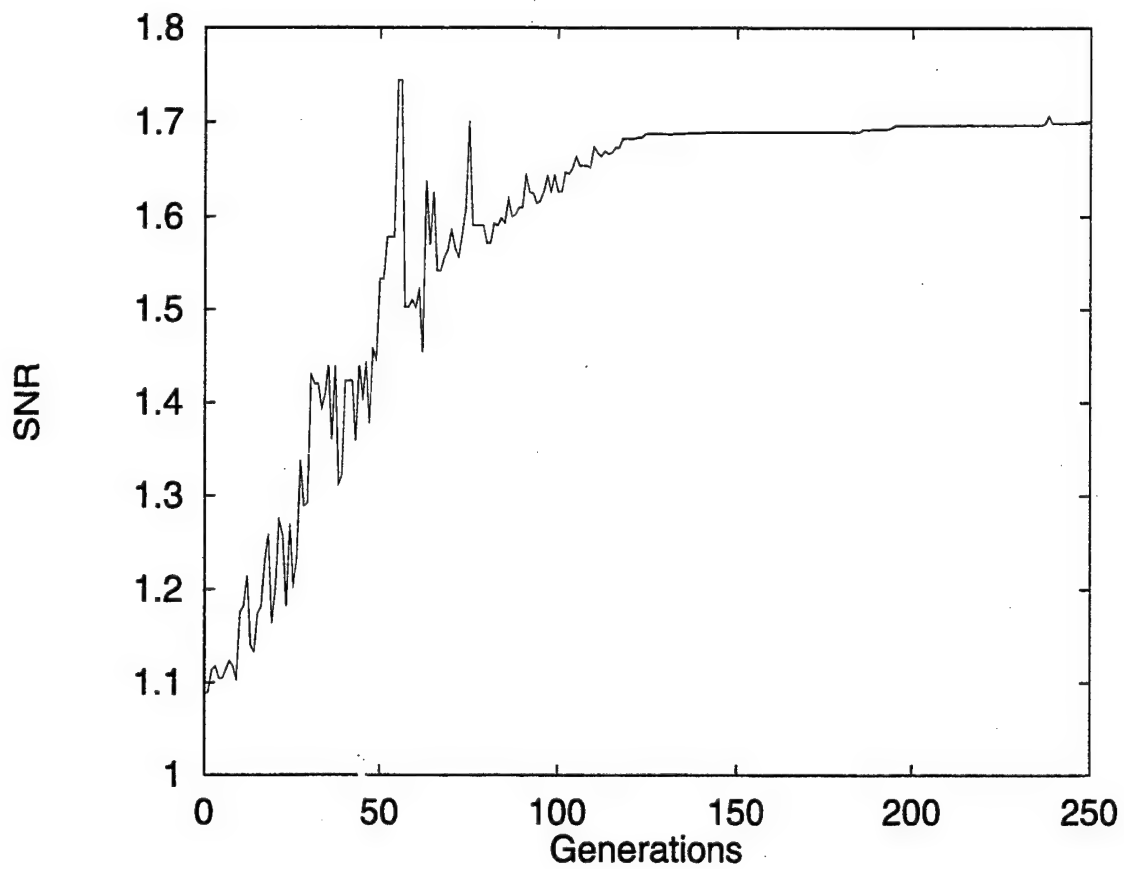
FIGURE 21 – SNR vs. number of generations for Trial E.
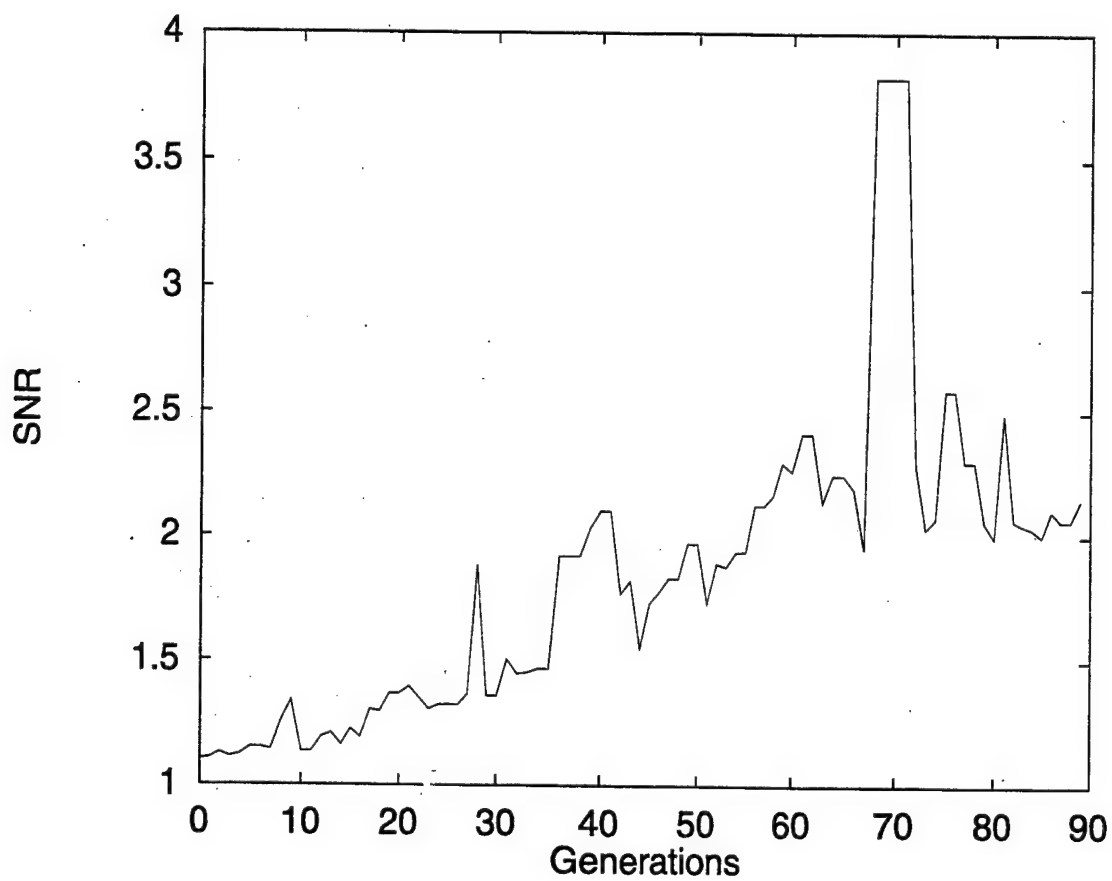
FIGURE 22 – SNR vs. number of generations for Trial F.

(a) Fractal Error
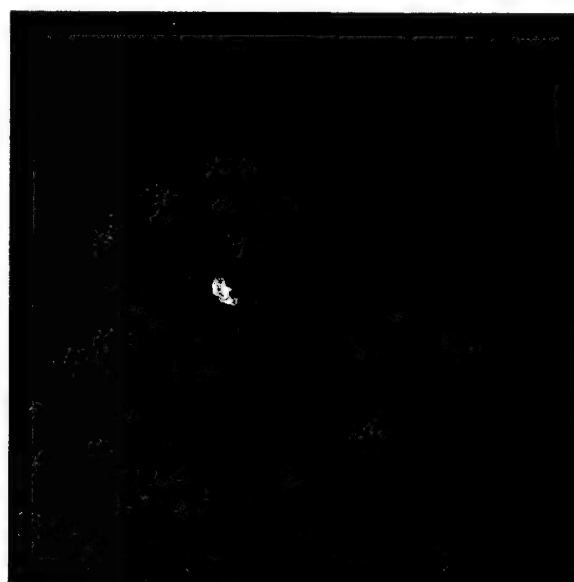
(b) GAFE

FIGURE 23 – A comparison of GAFE vs. fractal error for the Washington, DC image.
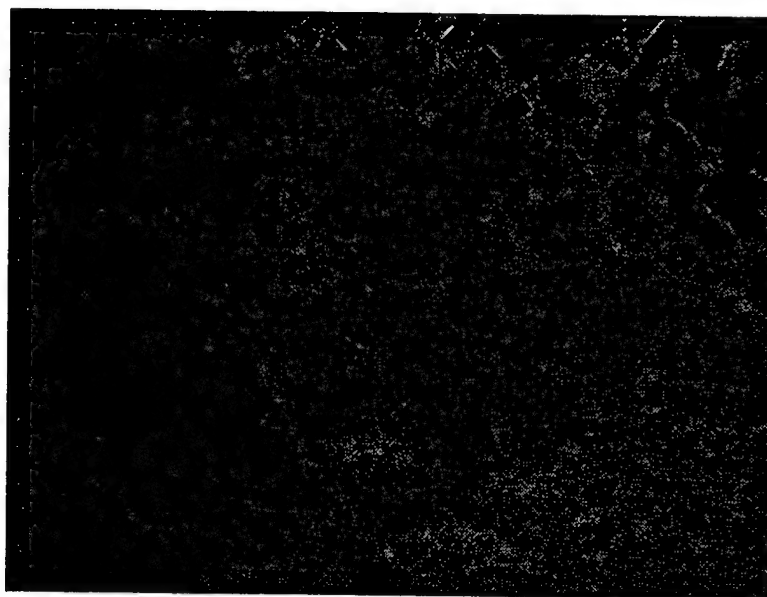
(a) Fractal Error
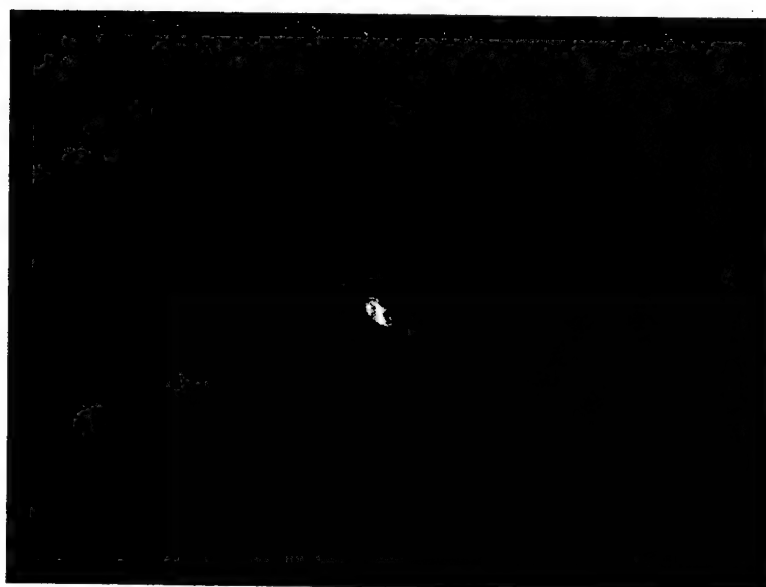
(b) GAFE

FIGURE 24 – A comparison of GAFE vs. fractal error for the SAR image of desert terrain.
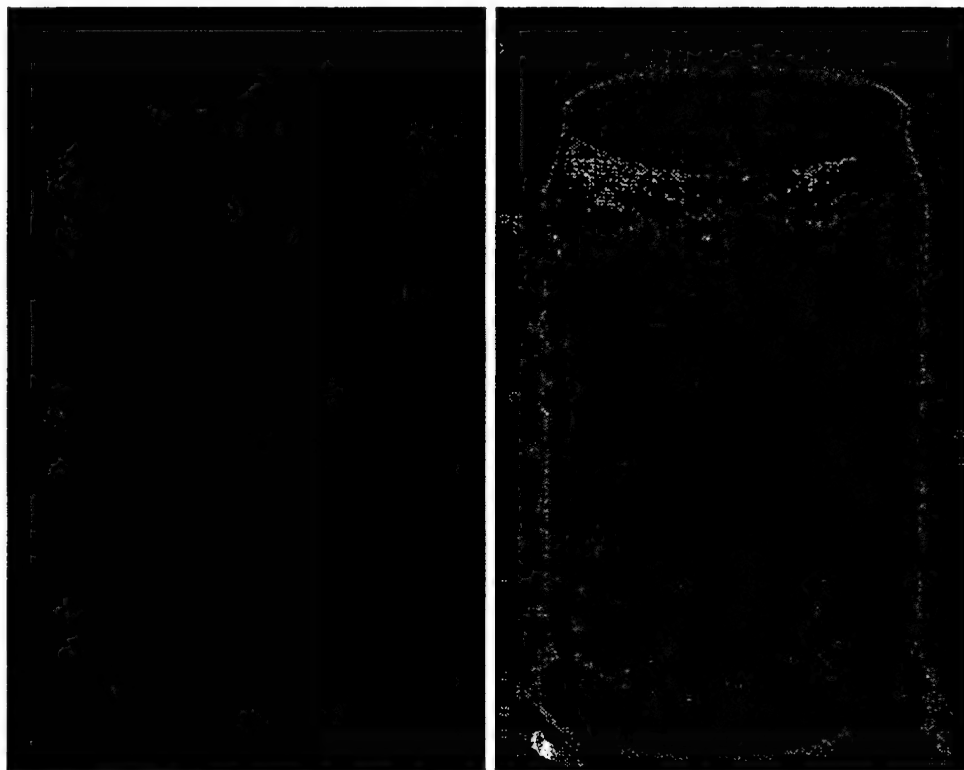
(a) Fractal Error   .   (b) GAFE

FIGURE 25 – A comparison of GAFE vs. fractal error for the Coke can image.
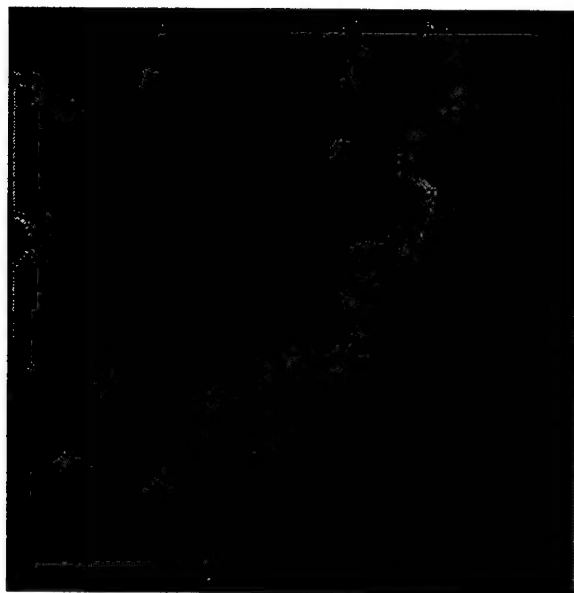
(a) Fractal Error
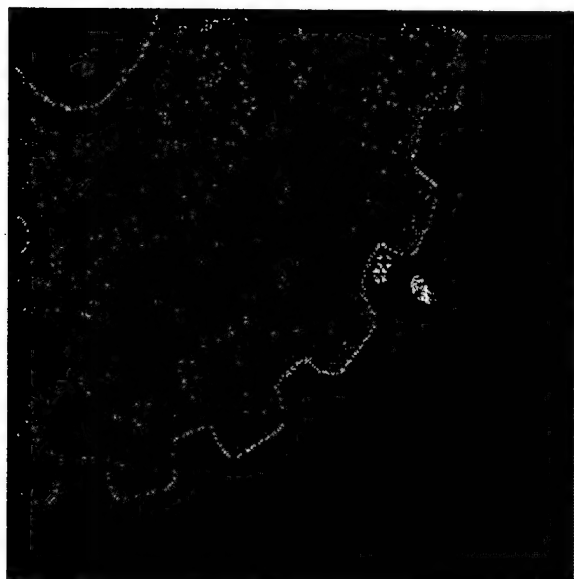
(b) GAFE

FIGURE 26 – A comparison of GAFE vs. fractal error for the Gear image.

CHAPTER V
TWO-DIMENSIONAL ENTROPIC SEGMENTATION

Segmentation is an important task in many image processing systems. Automatic target recognition may use segmentation to separate the desired target from background. Segmentation is often used to automatically highlight certain parts of the image that may be difficult to see with the human eye, such as cultural objects in synthetic aperture radar (SAR) imagery.

Thresholding gray scale values is the simplest method of segmentation. Thresholding implies that the object or objects in question have gray levels that are distinctly different from the background or other objects in the scene. For example, a black part on a white conveyor belt has distinctly different gray levels from the background it is against. The histogram for the gear image, shown in Figure 27, plainly shows the separation in gray-scale. Choosing a point somewhere between the large value at 0 and the peak at 190 will produce a segmented image as shown in Figure 28. However, examination of the histogram results in a second threshold point, between 130 and 160. Multi-point thresholding will produce the image shown in Figure 29. However, no useful information was gained from providing a second threshold.

Selecting thresholds can be grouped into two categories, local methods and global methods. Segmenting an entire image with a single value using the gray-scale histogram is a global method. Local methods partition an image into a group of sub-images and select a threshold point or set of threshold points for each of the sub-images. Global thresholding techniques are easy to implement, but some methods tend to be inaccurate, especially with complex images. The reader is encouraged to reference Sahoo et al. [55] and Wong and Sahoo [61] for excellent surveys of global
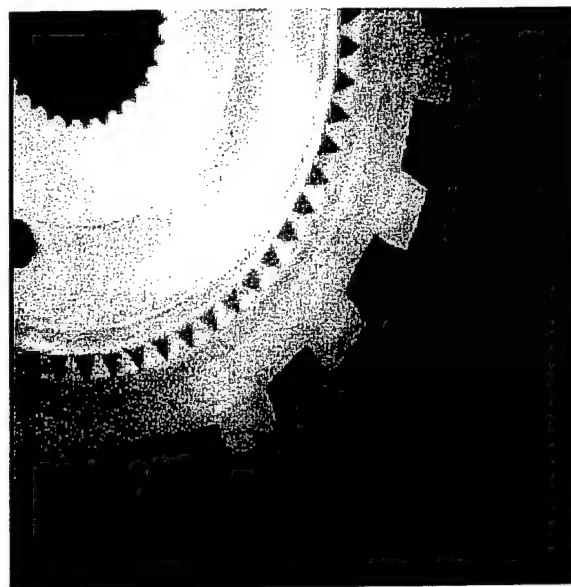
(a) Gear image
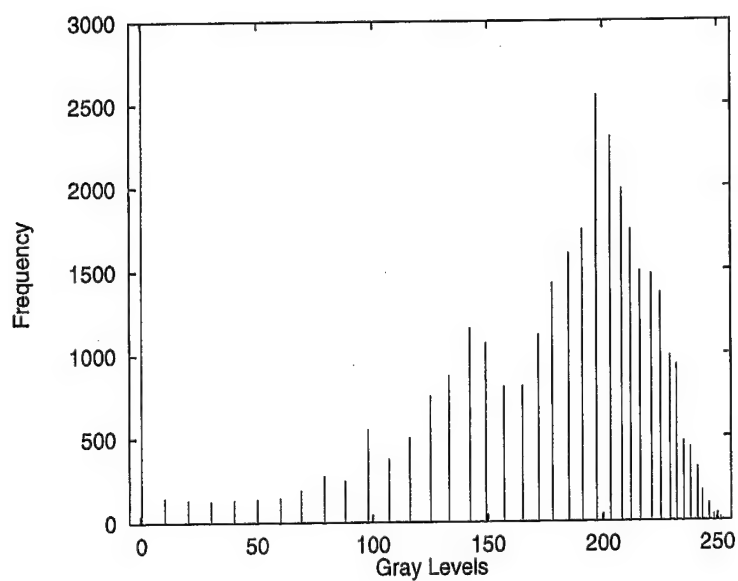
(b) Histogram of gear image

FIGURE 27 – Gear image and its resulting gray-scale histogram.
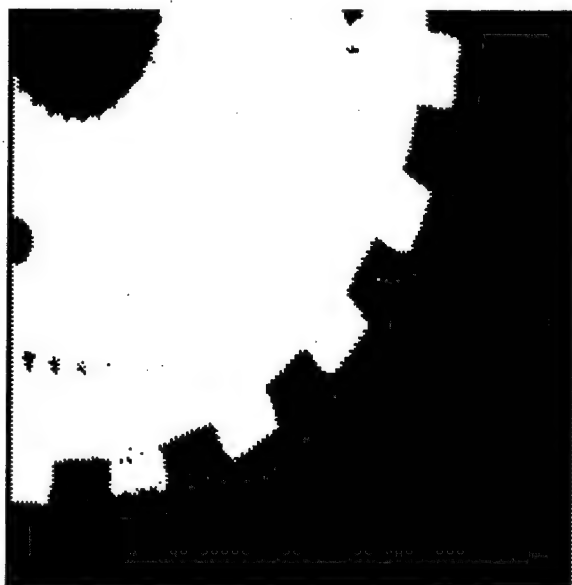
FIGURE 28 – Segmented gear image with a gray level threshold of 50.
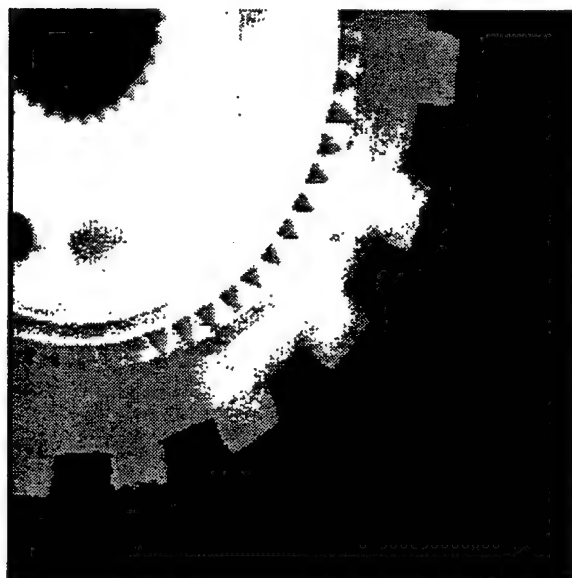
FIGURE 29 – Segmented gear image with gray level thresholds at 50 and 145.

thresholding methods using information theory.

This chapter presents a novel segmentation method after a technique presented by Sahoo, Slaaf and Albert [54]. This method finds an thresholding line using the information from a two-dimensional probability density function. Sahoo, Slaaf, and Albert proposed an entropic global thresholding method that selects a threshold point based on minimizing the difference between entropies of the object and the background distributions of the probability density function. The method presented here uses this thresholding point along with statistical information from the assumed data sets of the object and the background to produce a second point. The line that is produced from the two thresholding points is then used to separate the object data set from the background data set.

A.   General Theory

Sahoo, Slaaf, and Albert [54] published an excellent review of the theory behind their entropic thresholding method. It will be summarized here.

Thresholding typically uses the gray-level histogram of an image. Assuming a binary image, let $B = b_0, b_1$ be the pair of binary gray levels. B is the partition that separates the object from the background in the image. Mathematically, this partition is defined as

$$j(r,c) = \begin{cases} b_0 & \text{if } i(r,c) \leq t \\ b_1 & \text{if } i(r,c) > t. \end{cases} \tag{22}$$

where $i(r,c)$ is the gray-scale image, t is the gray-scale threshold, and $j(r,c)$ is the resulting binary image. In general, a thresholding method determines the value of t. Many methods use statistical information within the image to determine t. The gray-scale histogram can approximate the density function

$$h(x) = \text{Prob}[f(m,n) = x] \tag{23}$$

by using the expression

$$\widehat{h}(x) = \frac{g_x}{g_{total}},$$ (24)

where $\widehat{h}(x)$ is the resulting histogram of the image, g_x is the number of pixels with the gray- scale value x, and g_{total} is the total number of pixels in the image. g_{total} is equal to MN if the image is MxN.

There have been many different techniques proposed to estimate t. The maximum entropy sum method was proposed by Kapur, Sahoo, and Wong [35]. Their algorithm is based on the maximization of the information measure between the object and the background. Kapur, Sahoo, and Wong used the *a priori* entropy as the information measure. For a more detailed explanation of entropy, the reader should reference Chapter 15 of Papoulis [46].

Let $p_i = \widehat{h}(i) = n_i/MN$ define the estimate of the probability gray-level value. The *a priori* entropy is defined as

$$H_T = -\sum_{i=0}^{255} p_i \ln p_i$$ (25)

for the entire image. Assuming two classes of pixels, where b_0 denotes the class of "black" pixels (gray-scale value of 0) and b_1 denotes the class of "white" pixels (gray-scale value of 255), the *a priori* entropies for each class of pixels are

$$H_{b_0}(t) = -\sum_{i=0}^{t} \frac{p_i}{p(b_0)} \ln \frac{p_i}{p(b_0)},$$ (26)

$$H_{b_1}(t) = -\sum_{i=t+1}^{255} \frac{p_i}{p(b_1)} \ln \frac{p_i}{p(b_1)},$$ (27)

where

$$p(b_0) = \sum_{i=0}^{t} p_i$$ (28)

$$p(b_1) = \sum_{i=t+1}^{255} p_i.$$ (29)

Note that $p(b_0) + p(b_1) = 1$. Kapur, Sahoo, and Wong define the information between

the two classes to be

$$\Phi(t) = H_{b_0}(t) + H_{b_1}(t) = \ln[p(b_0)p(b_1)] + \frac{H_T}{p(b_0)} + \frac{H_T - H(t)}{1 - p(b_1)}, \qquad (30)$$

where $H(t)$ is defined as

$$H(t) = -\sum_{i=0}^{t} p_i \ln p_i. \qquad (31)$$

Kapur, Sahoo, and Wong selected the threshold to be the value of t^* at which $\Phi(t)$ is maximum. They noted that since some p_i will be very small, care should be taken in evaluating the term $\ln p_i$.

In [54], Sahoo, Slaaf, and Albert define the optimal threshold t^* by minimizing the difference of the entropies $H_{b_0}(t)$ and $H_{b_1}(t)$. The function

$$E(t) = [H_{b_0}(t) - H_{b_1}(t)]^2 \qquad (32)$$

measures the difference in the *a priori* entropies of classes b_0 and b_1. Sahoo, Slaaf, and Albert pointed out that the smaller $E(t)$, the more homogeneous are b_0 and b_1. Thus, the global minima of $E(t)$ can be used to denote t^*. This is expressed as

$$t^* = \text{Arg min } E(t). \qquad (33)$$

Sahoo, Slaaf, and Albert show that their entropy crossover method tends to be more accurate in some applications then other entropic thresholding methods, such as in [35].

B.   2-D Extension Of The Entropy Crossover Method

This section will outline the 2-D extension of the entropy crossover method outlined in [54] as well as the novel technique after the proposed extension. Often the information gathered by the gray-scale histogram is not enough to accurately segment a given image. In these cases, it is not uncommon to incorporate additional information in the histogram. The additional information results in a 2-D histogram
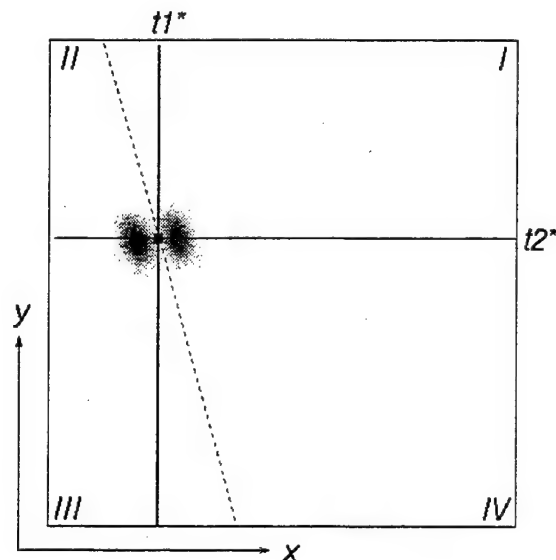
FIGURE 30 – The overall thresholding point for the 2-D histogram shown.

by computing the frequency distribution of the second feature and combining it with the gray-level histogram. This 2-D histogram represents a joint probability density function of the two features. The co-occurrence matrix is a commonly used 2-D histogram. Cooper [12] used a 2-D histogram made up of the gray-level values and fractal error.

Abutaleb [1] proposed that the optimal threshold point for a 2-D histogram can be found by locating the optimal thresholding point for each feature in the histogram. This will result in two separate thresholds, $t1^*$ and $t2^*$. When the threshold for each feature has been found, the two orthogonal lines segmenting the features divide the matrix into four quadrants. The intersection of the orthogonal lines produces the overall thresholding point $(t1^*, t2^*)$, as shown in Figure 30.

Brink [7] used the quadrants for segmentation. He segmented Quadrant II pixels as background and Quadrant IV pixels as foreground. However, he discarded the pixels located in Quadrants I and III, which may discard important information concerning the objects to be segmented. Instead of thresholding in quadrants, a

thresholding *line* in the 2-D histogram plane would provide better segmentation. Such a line is shown in Figure 30. Sahoo, Slaaf, and Albert recognized that the selection of a single point to classify the 2-D pdf may completely discard some sections of the desired distribution that reside near, but not in, the foreground or background quadrants. They suggested finding the optimal thresholding line by minimizing the difference in the entropy values on either side of a proposed line in the 2-D histogram plane, not unlike the method used in finding the optimal point in the 1-D histogram. However, this could prove to be a costly method, as there are approximately 3.9×10^5 lines that intersect a 256x256 histogram.

Instead of an exhaustive search of the possible thresholding lines, it is possible to calculate a thresholding line using the statistics of each quadrant. Let $(\overline{x}, \overline{y})_I$, $(\overline{x}, \overline{y})_{II}$, $(\overline{x}, \overline{y})_{III}$, and $(\overline{x}, \overline{y})_{IV}$ be the central moments for each of four quadrants respectively. Thus,

$$\overline{x} = \frac{\Sigma x_i}{p_{quad}} \tag{34}$$

$$\overline{y} = \frac{\Sigma y_i}{p_{quad}} \tag{35}$$

where p_{quad} is equal to the total number of points in a given quadrant. Figure 31 shows two separable classes in a 2-D histogram, and the central moment for each quadrant is computed. The quadrant with the most points is determined, and its central moment is used with the optimal threshold to form a line. This corresponds to Quadrant II in Figure 31. This resulting line is perpendicular to the desired thresholding line, as shown in Figure 32. The algorithm is summarized in Table 9.

C. Results

The segmentation algorithm was first tested on random data from different Gaussian distributions. Figures 33, 34, and 35 show each set of Gaussian data, the optimal thresholding point, the central moment, and the optimal thresholding line.

TABLE 9

Automatic 2-D Entropic Segmentation Algorithm

1.	Calculate the optimal thresholding point $(t1^*, t2^*)$ as described in Sahoo, Slaaf, and Albert [54].
2.	Divide the 2-D histogram into quadrants about $(t1^*, t2^*)$.
3.	Determine the quadrant with the largest number of points and calculate its central moment, $(\overline{x}, \overline{y})$.
4.	Compute the line, $l(x, y)$ that intersects $(t1^*, t2^*)$ and $(\overline{x}, \overline{y})$.
5.	Define $L(x, y)$ to be the line perpendicular to $l(x, y)$.
6.	Segment the image using $L(x, y)$ as the optimal thresholding line.

The thresholding line provides nearly error-free segmentation in each case.

The Alameda, CA aerial image, the Washington, DC aerial image, the SAR desert terrian image, and the suburban aerial image were segmented using the gradient values and fractal error as features. The results of each segmentation are shown in Figures 36 through 39. The resulting histograms, optimal thresholding points, central moments, and segmentation lines are shown in Figure 40. Man-made features and edges are easily identifiable in the Alameda, CA and Washington, DC images. However, the specular noise in the SAR makes segmentation difficult. The resulting segmentation of the suburban aerial image is poor. While the edges of the cultural objects, such as buildings, are outlined, the segmented image is noisy making it difficult to visually interpret.

Cooper [12] used gray-scale values and fractal error as features in a 2-D statistical classifier to successfully segment certain features in aerial images. Using gray-scale values and fractal error as features, results were produced using the segmentation al-

gorithm on the Alameda, CA image, the Washington, the DC image, the SAR image, and the suburban aerial image. The results of this segmentation are shown in Figure 41 through 44, and the resulting histograms are shown in Figure 45. The DC image produces poorly segmented results. It is likely that the information needed in the DC image, such as man-made objects or edges, could not be separated with a single linear curve. Perhaps a piece-wise linear curve or a higher-order curve would provide a better segmentation using gray-scale value and fractal error. However, the segmentation results for the Alameda, SAR, and suburban images are very good. The cultural objects are highlighted and the natural terrian is classified as background. There is some misclassification of pixels in the SAR due to specular noise, but not enough to hinder visual correlation.

Figure 46 shows ground truth for the suburban aerial image. For the resulting segmentation image using gradient and fractal error, shown in Figure 39, the misclassification of pixels is 25 %. However, for the resulting segmentation image using gray levels and fractal error, shown in Figure 44, the misclassification is only 13 %. This error may be due, in part, to the occlusion of houses and other cultural objects by the trees in the image.

Cooper compared the results from classification using a modified busyness operator [12] with the results from classification with fractal error. Figure 47 illustrates this modified busyness operator with the suburban aerial image. Figure 48 shows the 2-D entropic segmentation of the suburban aerial image using as features the modified busyness operator and gray levels. The misclassification of pixels with the modified busyness operator is 22.2 %. However, if the normalized busyness image is used instead, the misclassification is 17 %. The resulting images do not provide the accuracy that fractal error did with this particular segmentation method.

The segmentation algorithm was accurate for each set of Gaussian data. However, success on real-world images is dependent on an intelligent choice of features.
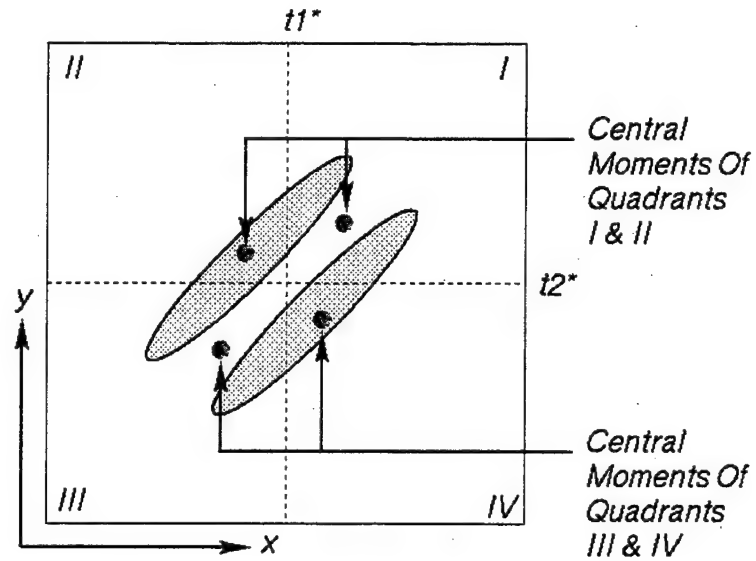
FIGURE 31 – Two separable classes with the optimal threshold point and the four quadrant's central moments.

Future research should focus on a study of segmentation with different features. Such a study may provide an indication of which features may be more effective than others using the segmentation algorithm.
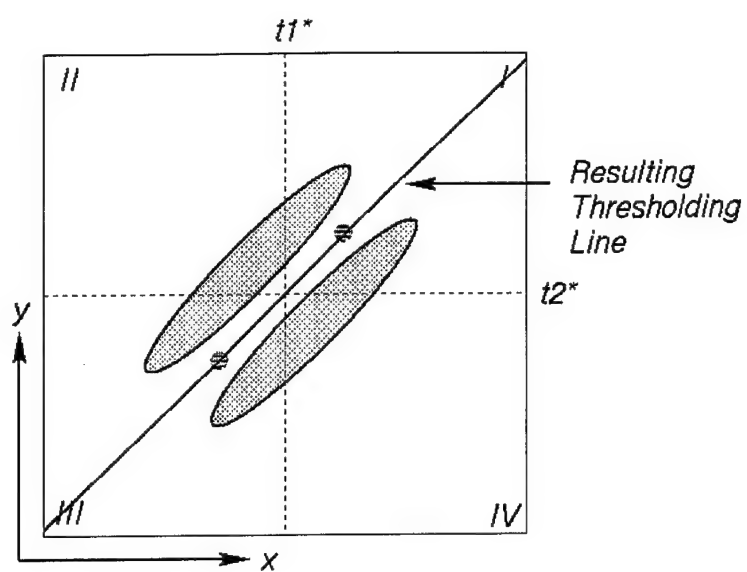
FIGURE 32 – Two separable classes and the resulting optimal threshold line.
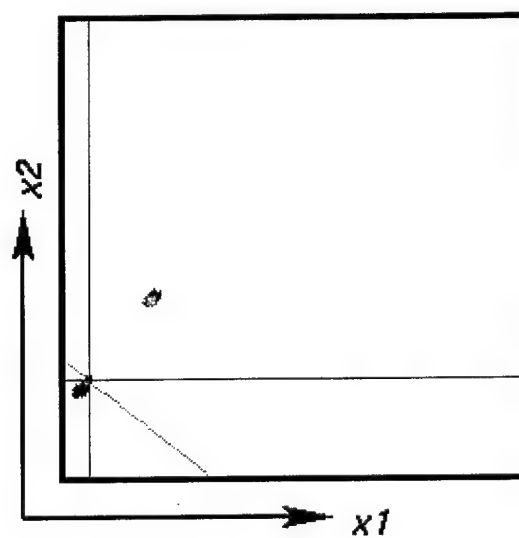
FIGURE 33 – Example 1–Two Gaussian data sets and the resulting threshold line.
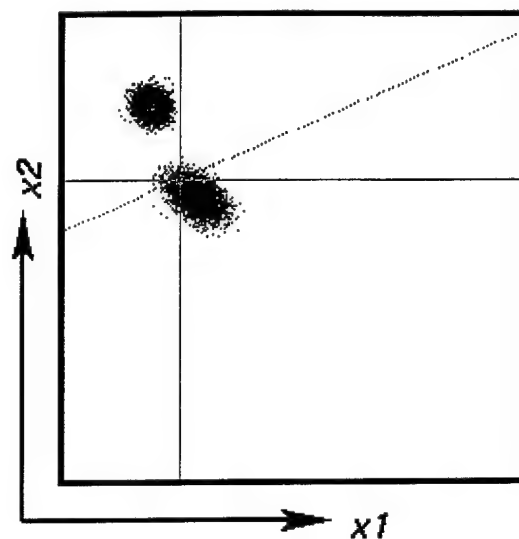
FIGURE 34 – Example 2–Two Gaussian data sets and the resulting threshold line.
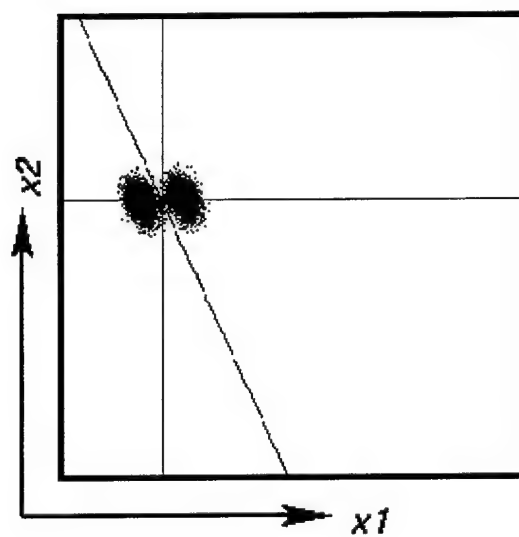
FIGURE 35 – Example 3–Two Gaussian data sets and the resulting threshold line.
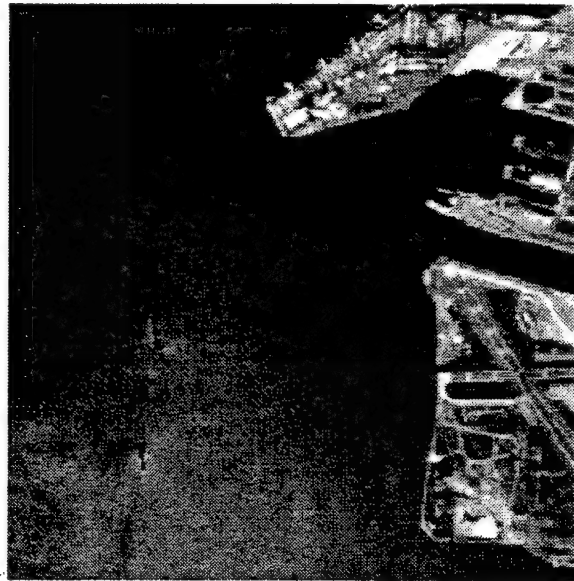
(a) Original Image

(b) Segmented Image

FIGURE 36 – Aerial image of Alameda, CA segmented using gradient and fractal error as features.
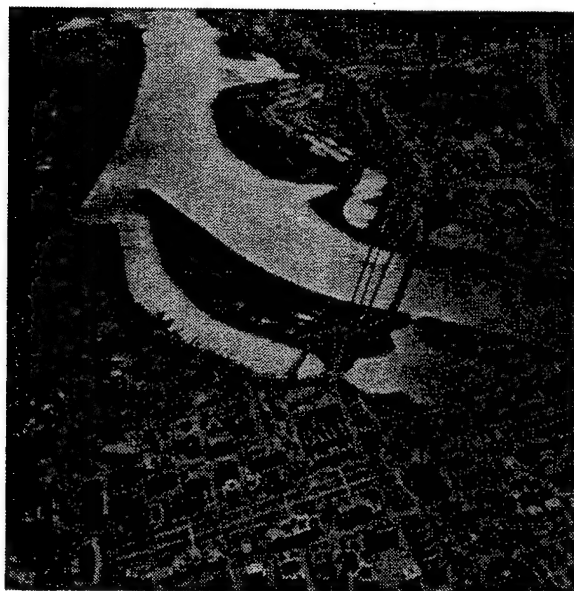
(a) Original Image

(b) Segmented Image

FIGURE 37 – Aerial image of Washington, DC segmented using gradient and fractal error as features.
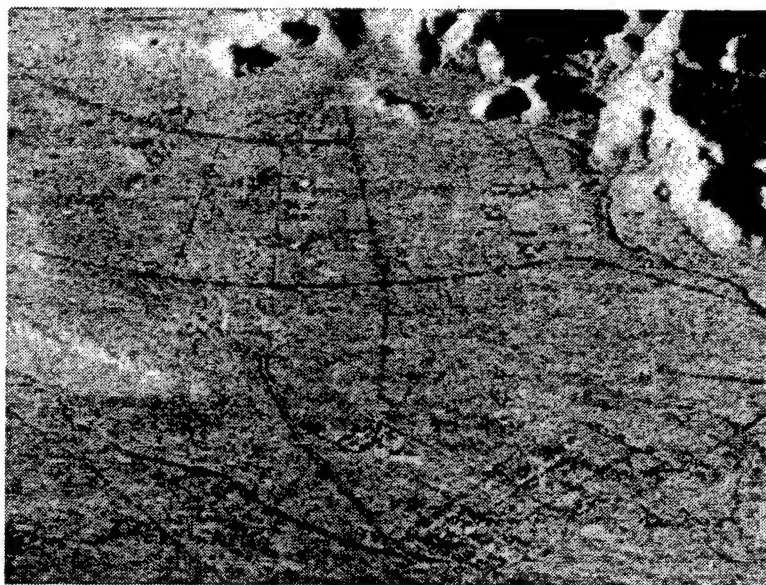
(a) Original Image

(b) Segmented Image

FIGURE 38 – SAR image of desert segmented using gradient and fractal error as features.

(a) Original Image
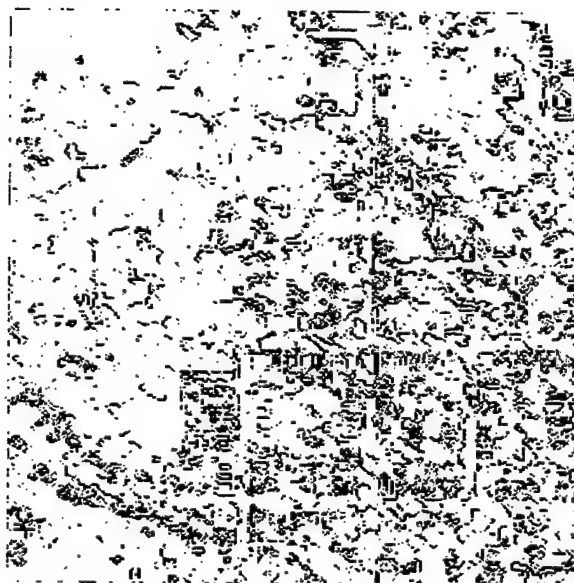
(b) Segmented Image

FIGURE 39 – Aerial image of suburban area segmented using gradient and fractal error as features.
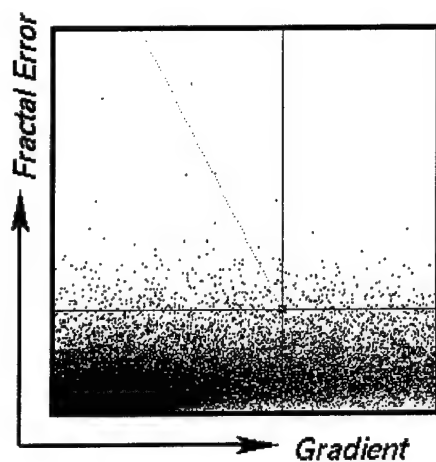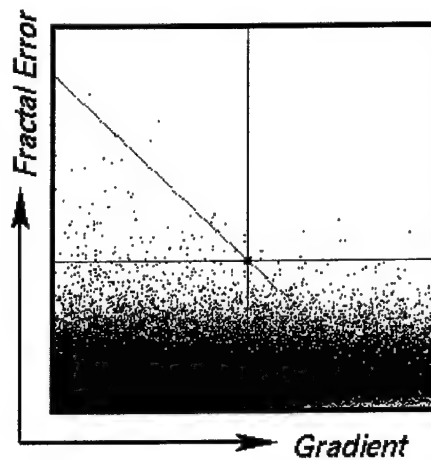
(a) Alameda, CA Image

(b) Washington, DC Image
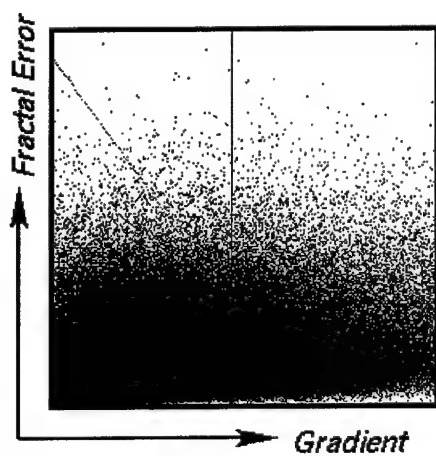
(c) SAR Image
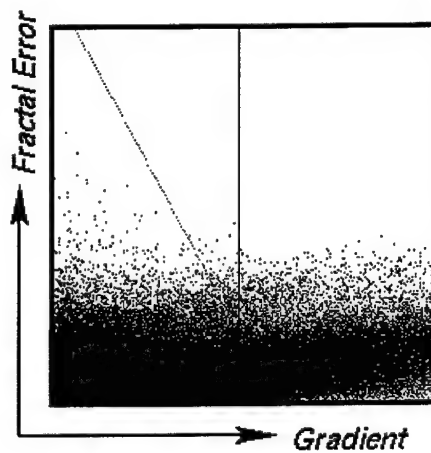
(d) Suburban Image

FIGURE 40 – 2-D Histograms for each of the images shown in Figures 36 through 39 with the resulting threshold lines.

(a) Original Image

(b) Segmented Image

FIGURE 41 – Aerial image of Alameda, CA segmented using gray levels and fractal error as features.
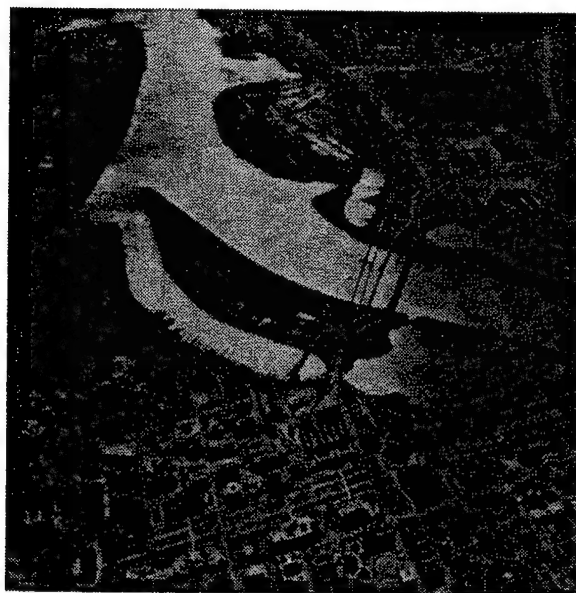
(a) Original Image

(b) Segmented Image

FIGURE 42 – Aerial image of Washington, DC segmented using gray levels and fractal error as features.

(a) Original Image

(b) Segmented Image

FIGURE 43 – SAR image of desert terrian segmented using gray levels and fractal error as features.

(a) Original Image

(b) Segmented Image

FIGURE 44 – Aerial image of suburban area segmented using gray levels and fractal error as features.
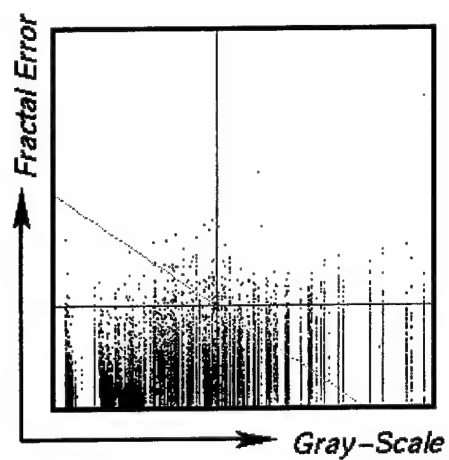
(a) Alameda, CA Image
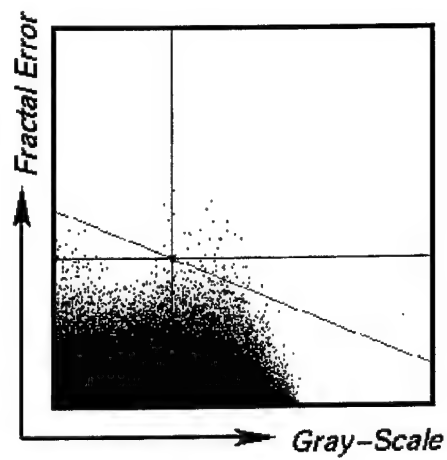
(b) Washington, DC Image
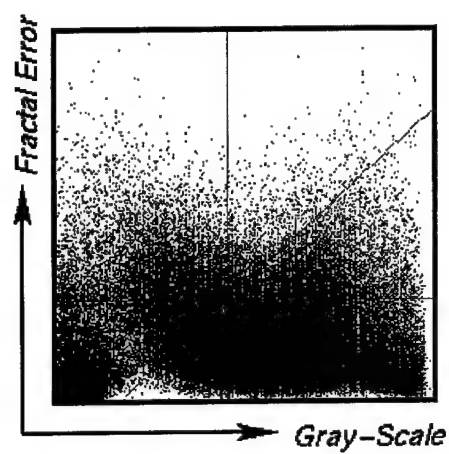
(c) SAR Image
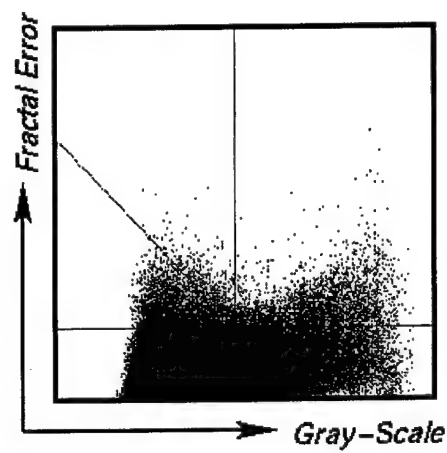
(d) Suburban Image

FIGURE 45 – 2-D Histograms for each of the images shown in Figures 41 through 44 with the resulting threshold lines.

FIGURE 46 – Ground truth for the suburban aerial image shown in Figures 39 and 44.
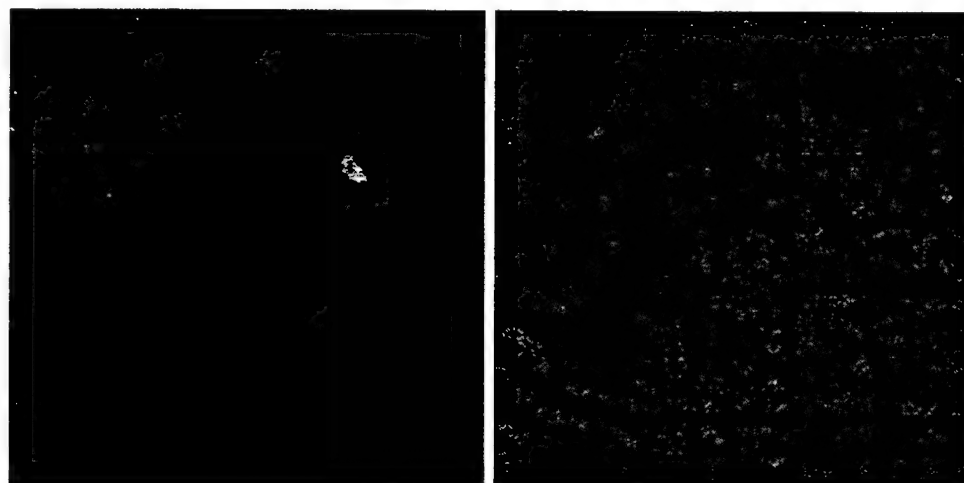
(a) Busyness Image

(b) Normalized Busyness Image

FIGURE 47 – Resulting busyness images using the aerial image of suburban area.

(a) Busyness
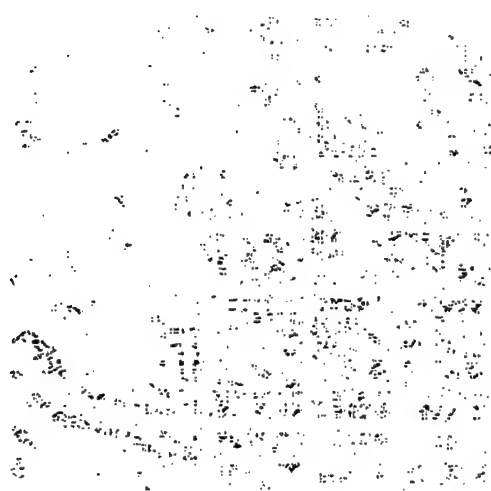
(b) Normalized Busyness

FIGURE 48 – Aerial image of suburban area segmented using gray levels vs. busyness and gray levels vs. normalized busyness as features.

CHAPTER VI
FRACTAL ERROR-BASED EDGE DETECTION

As previously mentioned, edge detection is a two-phase process. The first step is edge enhancement, or the process of locating pixels in the image that may be considered an edge. Edge linking attempts to link edges that might be broken. Edge enhancement and linking using fractal error are described in this chapter.

A. Edge Enhancement

Edge enhancement is the process of detecting pixels that may be part of a step edge in the scene. Fractal error is a useful measure for enhancing edges, which may be boundaries between textures or shading. These boundaries do not fit well in the fBm model, therefore, normalized fractal error provides a likelihood, a probability that a pixel may belong to a boundary edge. Thus, the higher the fractal error value (and thus, the poorer the fit in the fBm model), the higher the likelihood that the pixel is part of the edge map. Edge pixels are assigned by simple thresholding, and false edges are eliminated by adjusting a thresholding constant, σ.

Table 10 outlines the enhancement algorithm. Figure 49 shows the gear image whose edges have been enhanced with this algorithm. The edges are barely detectable in the image when the threshold σ was chosen too high, such as the image in Figure 49 (a). When the threshold is too low, the edges are enhanced along with the noise in the image, such as the scene in Figure 49 (d). Further study is needed to determine an optimal threshold.

TABLE 10

Edge Enhancement Algorithm

1. Calculate fractal error for the image.

2. Normalize fractal error.

3. Specify the thresholding constant σ.

4. Choose candidate edge pixels if fractal error is greater than σ.
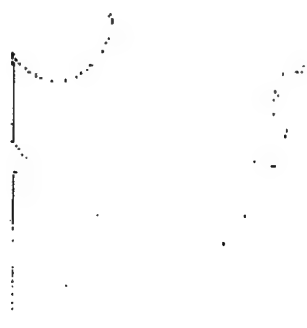
(a) $\sigma = 0.5$ (b) $\sigma = 0.3$
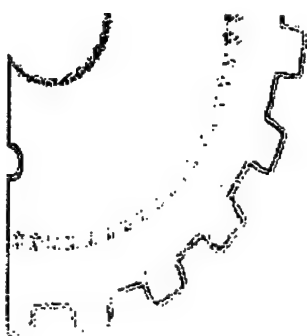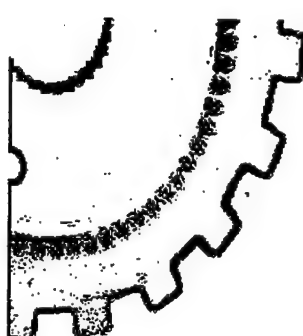
(c) $\sigma = 0.25$ (d) $\sigma = 0.1$

FIGURE 49 – Edge enhancement with the gear image for various σ thresholds.

= Candidate Edge Pixel
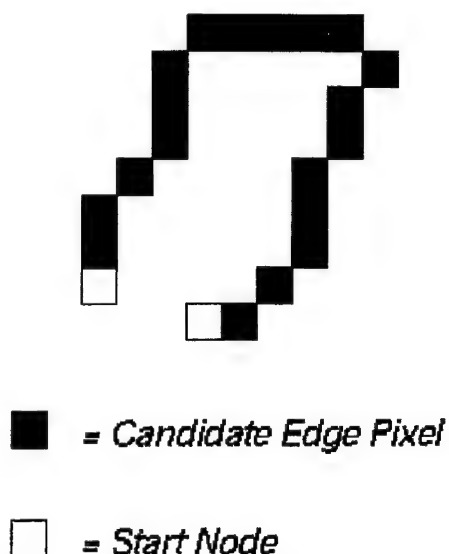
= Start Node

FIGURE 50 – Candidate edge pixels, with start nodes defined.

B.   Edge Linking

Edge linking involves connecting edges that are disconnected either by a lack of change in gray level values, by noise, or by occlusion. Broken edges may occur with fractal error edges because of a lack of dissimilarity between two textures or by noise masking the edges by assuming a sort of false self-similarity.

Linking requires defining start nodes, a fitness measure for each possible path, and stop criteria. Edge pixels with no neighboring edge pixel or one neighboring edge pixel are designated as start nodes for the linking algorithm. An illustration of start nodes is shown in Figure 50.

Farag and Delp [19] suggested that the number of edge paths to be searched can be reduced significantly by restricting each component of the path to no more than a 45° turn. This reduces the very large set of edge paths of length five to twenty-four unique paths. They are shown in Figure 51. The best path is then found by exhaustive search through all twenty-four paths. The fitness of each path is defined
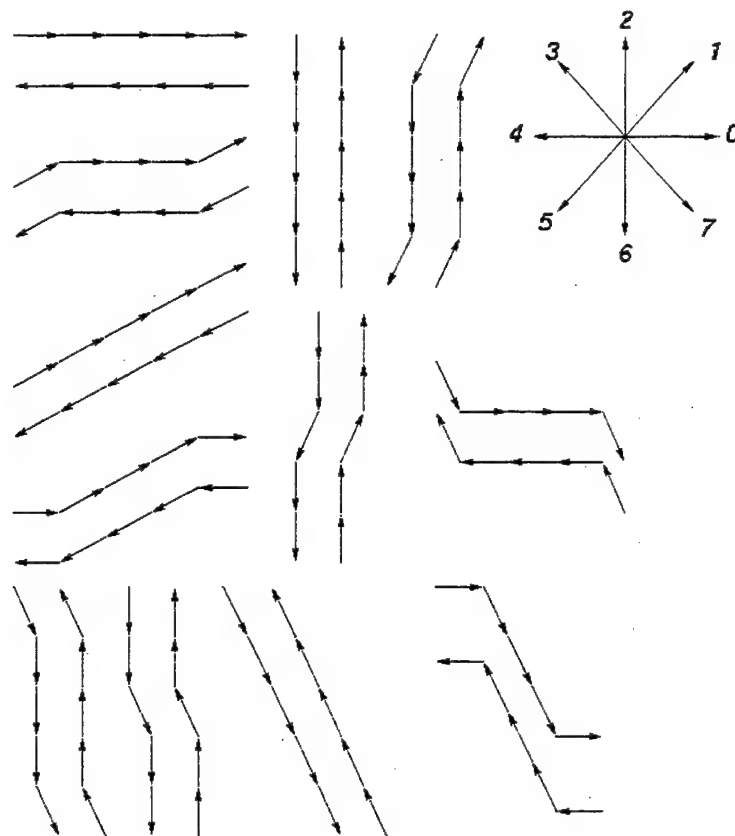
FIGURE 51 – The twenty-four unique edge paths suggested by Farag and Delp [19].
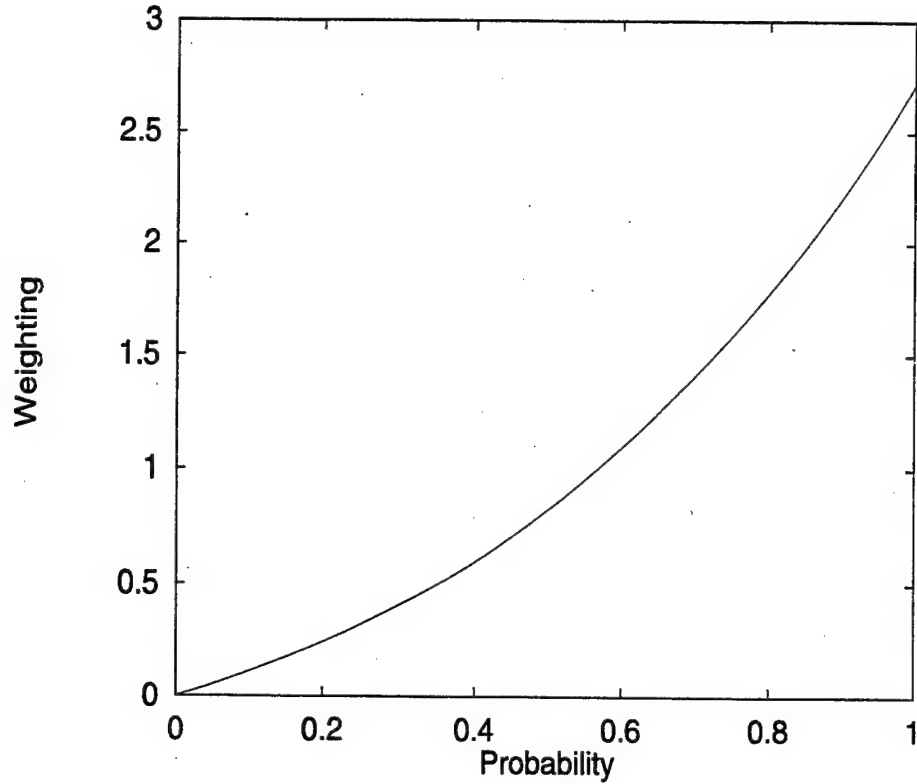
FIGURE 52 – Weighting function $w(p_i)$.

as

$$F = \sum_{i=0}^{4} w(p_i), \tag{36}$$

where p_i is the possibility of a particular pixel being an edge, that is, its normalized fractal error value, and $w(p_i)$ is the weighting function

$$w(p_i) = p_i \exp(p_i). \tag{37}$$

Figure 52 illustrates the weighting function. Pixels with high likelihood of being part of the edge path have an exponentially higher weight than those who are not likely to be part of the path. The edge path with the highest value of F is chosen to be the best path. The end of the path is marked as a start node and the edge tracing continues.

The tracing continues for a particular edge path until the path encounters an already defined edge pixel or another start node. The tracing is also terminated

if the fitness for a chosen path is less than a particular user defined threshold τ. This threshold prevents the algorithm from straying from the desired edge boundary, especially in noisy images that may have deceivingly high fractal error values in or around an edge boundary. If a particular path has a fitness that is less than τ, that path is discarded and the tracing ceases. The search begins again for another start node.

Once the best path is chosen from the start node, the final direction of the best path is used as a criteria for defining the next best path. For example, if the best path after a start node ends with an easterly direction, defined as direction 0 in Figure 51, then the next best edge must either start in an easterly, northeasterly, or southeasterly (0, 1, or 7, respectively) direction. Therefore, if the directions are numerically assigned as in Figure 51, then the three candidate path directions for a next path are expressed as

$$C_0 = (P_D - 1)\%8 \tag{38}$$

$$C_1 = P_D \tag{39}$$

$$C_2 = (P_D + 1)\%8 \tag{40}$$

where C_0, C_1, and C_2 are the three candidate edge directions, P_D is the final direction of the previous path, and % defines the modulus operation. Table 11 outlines the entire edge linking process.

C.   Results

A synthetic image was used to test the edge detection algorithms. Figure 53 shows the test image with no noise present. The enhancement threshold σ was equal to 0.25 in each image. The edge linking threshold was allowed to vary for different trials. Each trial is shown in Figure 54. The edge detection is consistently good for each value of τ.

TABLE 11

Edge Linking Algorithm

1. Determine possible edge pixels with edge enhancement algorithm.

2. Determine start nodes.

3. For each start node, search through possible paths for best edge path.

4. Stop search if chosen path intersects a previously defined edge pixel or another start node, or if the chosen path's fitness is less than the user-defined threshold τ.

5. If chosen path's is less than τ, discard path.

6. Otherwise, calculate the three candidate directions for next path and determine best fit edge.

7. Continue steps 3-6 until all start nodes are linked.
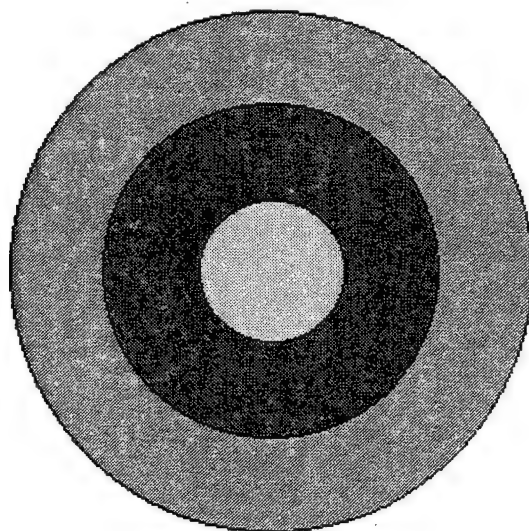
FIGURE 53 – Noiseless synthetic image used to test fractal error edge detection.

(a) $\tau = 1.5$

(b) $\tau = 1.0$

(c) $\tau = 0.75$

(d) $\tau = 0.5$
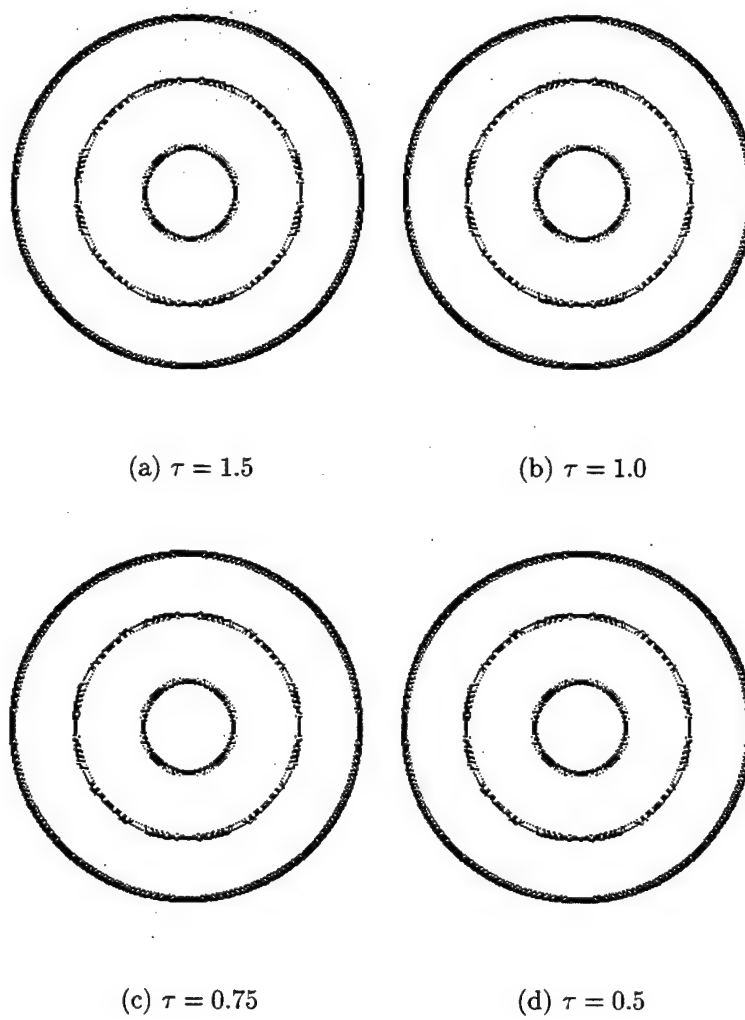
FIGURE 54 – Results of fractal error edge detection with test image in Figure 53.
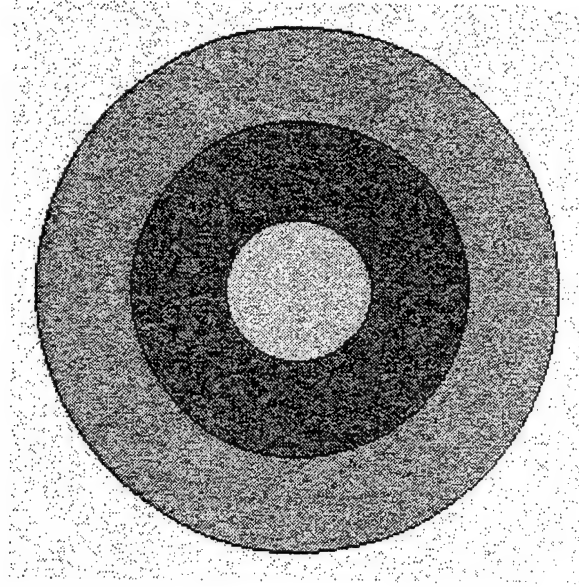
FIGURE 55 – Noisy synthetic image used to test fractal error edge detection.

To test the robustness of the algorithm, the synthetic image was corrupted with Gaussian or "white" noise. Figure 55 shows the synthetic image corrupted with noise. The algorithm was executed over this image with $\sigma = 0.25$ and $\tau = 1.5, 1.0, 0.75$ and 0.5, as shown in Figure 56. It is clear that the edge paths wander from the edge boundaries when the path cutoff τ is chosen poorly. However, it is not clear what an acceptable choice of τ should be. If τ is too large, gaps in the edge boundary will not be linked. On the other hand, if τ is too small, false edges will be present.

To compare the fractal error edge detection to another well-established edge method, the noiseless test image and the noisy test image had their edges detected by the fractal error edge detection as well as the Canny edge operator. The Pratt figure of merit [30] is a numerical measure of the accuracy of an edge detector. It is expressed as

$$F = \frac{1}{I_L} \sum_{i=1}^{I_A} \frac{1}{1 + \alpha d^2}, \tag{41}$$

where $I_L = \max(I_L, I_A)$, I_I and I_A represent the number of ideal and actual edge points, α is a scaling constant, and d is the separation distance from an actual edge

(a) $\tau = 1.5$

(b) $\tau = 1.0$

(c) $\tau = 0.75$

(d) $\tau = 0.5$
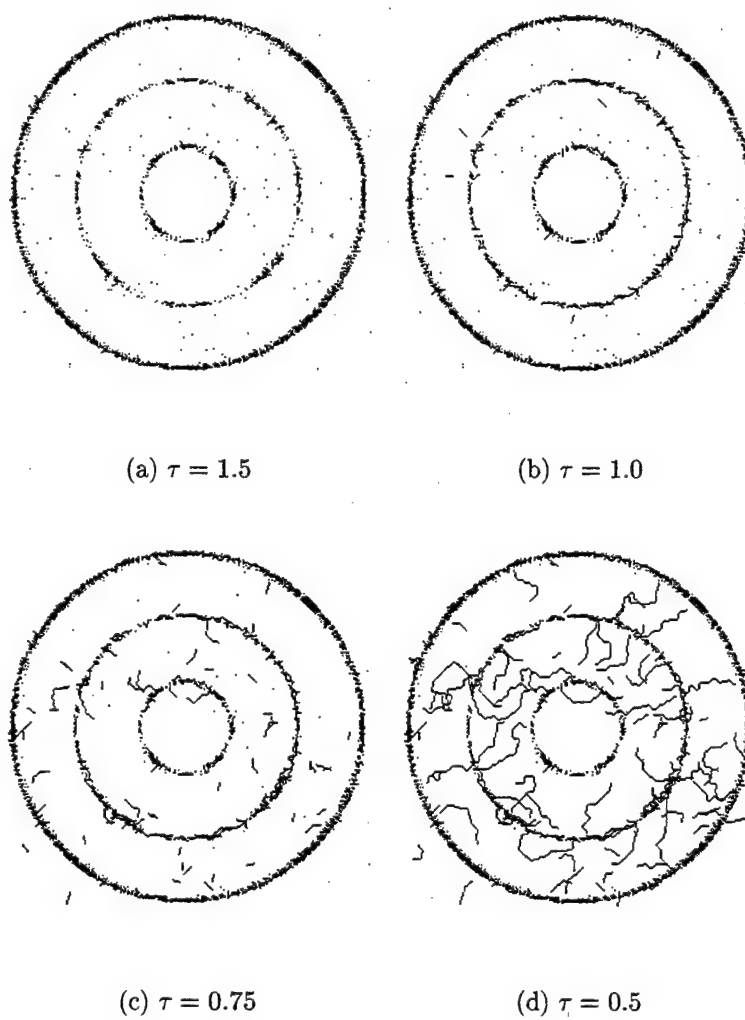
FIGURE 56 – Results of fractal error edge detection with the noisy test image in Figure 55.

point to the nearest ideal edge point. For ideal edge detectors, F will equal 1. For most edge detectors, the Pratt figure of merit will decrease as the signal-to-noise ratio (SNR) decreases. Jansing, Allen, and Chenoweth [32] used the Pratt figure of merit to show that fractal error edge enhancement was more robust in a noisy environment than the Canny edge operator.

Figure 57 shows the edges of the noiseless test image (Figure 53) and the edges of the noisy test image (Figure 55). Both images were smoothed with a Gaussian kernel ($\sigma = 1.0$) and thresholded at the gray-scale value 180. The Pratt figure of merit for these two images was $F = 0.9952$, with the edges from the noiseless test image defining the ideal edges. Using the scene from Figure 54 (a) as the ideal edges for fractal error edge detection, the Pratt figure of merit was computed for the scene in Figure 56 (a), producing $F = 0.9656$. This illustrates that the fractal error edge detection algorithm is comparable to the Canny edge operator in performance. There may be some improvement in the Pratt figure of merit for the fractal error edge detection algorithm, if the noisy image was smoothed before the fractal error edges were detected, like the smoothing performed in the Canny edge operator.

The gear image was also used to illustrate the usefulness of this fractal error edge detection. Figure 58 shows the detected edges of the gear scene with a constant $\sigma = 0.25$ and varying τ. In each case, the gear was outlined satisfactorily, with the exception of a tooth in the lower left section of the part. This may be corrected by lowering the value of σ, but at the risk of amplifying noise in the image. Decreasing the value of τ may also promote the linking of the edges on either side of the gap, but may increase the probability of "wandering" edge paths, which lead to false edges. Further study is needed to determine consistent selection of the optimal σ and τ thresholds.

There are two serious limitations to this edge detection: the time needed to search through the edge paths and the lack of information in choosing an optimal
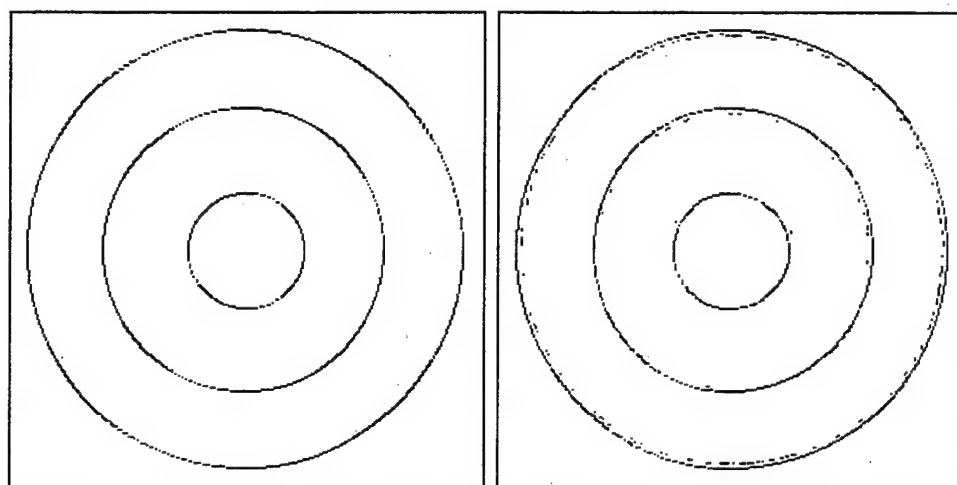
(a) Noiseless Test Image   (b) Noisy Test Image

FIGURE 57 – Edge detection of the noiseless and noisy test images using the Canny edge operator.

τ. These issues should be addressed in future work. The exhaustive search could be replaced by an intelligent search, such as an A^* search (used by Farag and Delp in their edge linking algorithm) or a genetic search. The selection of τ could be eliminated completely by modifying F of Equation 36 so that selected edges would have easily distinguishable fitness values. For example, poorly fitting edges would have fitness values near zero and well-fit edges would possess fitnesses values around 1. Then, intelligent selection of edges could be automatic and not require the assistance of the user.

(a) $\tau = 1.5$

(b) $\tau = 1.0$

(c) $\tau = 0.75$

(d) $\tau = 0.5$
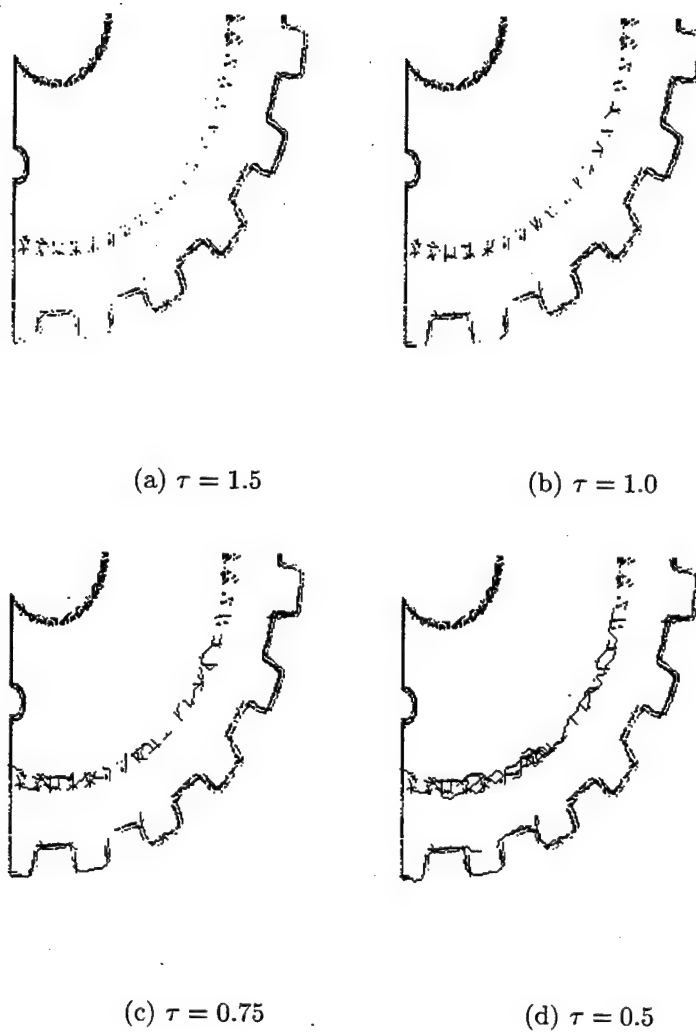
FIGURE 58 – Results of fractal error edge detection with the gear image.

CHAPTER VII
CONCLUSIONS

Presented in this work were three methods relating to feature and edge extraction using fractal error. The first method, a genetic approximation to fractal error called GAFE, was shown to adequately mimick the original fractal error algorithm with an average SNR value of 3. This method evolved a set of weights in a fixed mathematical structure. The best set of weights is used, along with the mathematical structure, to produce the approximation to fractal error.

A novel 2-D entropic segmentation method was introduced. This method was shown to be very accurate using Gaussian data. Given a set of features that produce two unique or nearly unique classes, this method should automatically segment an image with little error. Test images were shown using gradient and fractal error, as well as gray levels and fractal error, as the features in the segmentation.

Edge detection using fractal error as a likelihood measure was shown to be accurate for synthetic and real-world images. Using the Pratt figure of merit, the fractal error edge detection algorithm was shown to be competitive with the Canny edge operator. However, the exhaustive search for possible edge paths in the edge linking process created a computationally intensive algorithm.

Fractal error should be considered an extremely useful measure with numerous applications. For example, the fractal error metric could be useful in the analysis of images recently transmitted from the *Pathfinder* and *Sojourner* mission on Mars. Automatic target recognition and detection of cultural objects in aerial imagery can be performed accurately with fractal error. The detection of self-similar blocks of pixels using fractal error could provide a quick, effective method of image compression,

compressing those self-similar blocks and keeping those block which are not self-similar in nature. The following is a brief list of future research areas that utilize fractal error.

- An quantitative study of different window sizes and shapes in the calculation of fractal error is needed.

- The development of an approximation to fractal error using genetic programming is needed. This approach would allow the mathematical structure of the approximation to evolve, rather than just a set of weights to a fixed mathematical structure. This evolved structure may provide a more accurate approximation of fractal error.

- An quantitative investigation of fractal error as a texture measure would be revealing, comparing fractal error to other traditional texture features, such as Haralick's co-occurrence features [23].

- The specular noise of unfiltered SAR images was shown to be problematic in segmentation with fractal error. Thus, a study of cultural object detection on filtered SAR images using fractal error is required.

- The choice of the constants σ and τ outlined in the fractal error edge detection algorithm is undefined and heuristic. Research to define optimal selection of σ and τ would provide a more automatic edge detection algorithm.

- A major disadvantage to the fractal error edge detection algorithm is computationally expensive. Development of a fast search method, such as a genetic or simulated annealing search, may improve the processing time of edge detection using fractal error.

REFERENCES

[1] A. S. Abutaleb. Automatic thresholding of gray-level pictures using two- dimensional entropy. *Computer Vision, Graphics and Image Processing*, 47:22–32, 1989.

[2] L. J. Bain and M. Engelhardt. *Introduction to Probability and Mathematical Statistics*. PWS-Kent, Boston, 1992.

[3] D. H. Ballard. Generalized hough transform for detecting arbitrary shapes. *Pattern Recognition*, 13:111–122, 1981.

[4] F. H. Bennett, J. R. Koza, D. Andre, and M. A. Keane. Evolution of a 60 decibel op amp using genetic programming. In *Proceedings of the 1st International Conference on Evolvable Systems: From Biology to Hardware*, pages 455–69, October 1996.

[5] V. Berzins. Accuracy of the Laplacian edge detector. *Computer Vision*, 27:195–210, 1984.

[6] M. La Brecque. Fractal applications. *Mosaic*, 17(4):34–48, Winter 1986/1987.

[7] A. D. Brink. Thresholding of digital images using two-dimensional entropies. *Pattern Recognition*, 25(8):803–808, August 1992.

[8] P. Brodatz. *A Photographic Album For Artists and Designers*. Dover, New York, 1966.

[9] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, PAMI-8(6):679–698, November 1986.

[10] J. J. Clark. Authenticating edges produced by zero-crossings. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, PAMI-11(1):43–57, January 1989.

[11] R. W. Conners. Towards a set of statistical features which measure visually perceivable qualities of texture. In *Proceedings of Pattern Recognition and Image Processing Conference*, pages 382–390, 1979.

[12] B. E. Cooper. *Fractional Brownian Motion For Representation Of Natural Image Texture*. PhD thesis, University of Louisville, 1994.

[13] B. E. Cooper, D. L. Chenoweth, and J. E. Selvage. Fractal error for detecting man-made features in aerial images. *Electronics Letters*, 30(7):554–555, 1994.

[14] L. Davis. *Genetic Algorithms and Simulated Annealing*. Morgan Kaufmann Publishers, Inc., London, 1987.

[15] H. Derin, P. A. Kelly, G. Vezina, and S. G. Labitt. Modeling and segmentation of speckled images using complex data. *IEEE Transactions on Geoscience and Remote Sensing*, 28:76–87, 1990.

[16] G. A. Edgar, editor. *Classics On Fractals*. Addison-Wesley, Reading, MA, 1993.

[17] P. H. Eichel, E. J. Delp, K. Koral, and A. J. Buda. A method for fully automatic detection of coronary arterial edges for cineangiograms. *IEEE Transactions on Medical Imaging*, MI-7(4):313–320, December 1988.

[18] K. Falconer. *Fractal Geometry–Mathematical Foundations and Applications*. John Wiley and Sons, Ltd., New York, 1990.

[19] A. A. Farag and E. J. Delp. Edge linking by sequential search. *Pattern Recognition*, 24(5):611–633, 1995.

[20] A. Gagolowicz. Visual discrimination of stochastic texture fields based upon second order statistics. In *Proceedings of the 5th International Joint Conference on Pattern Recognition*, pages 768–789, 1980.

[21] D. E. Goldberg. *Genetic Algorithms In Search, Optimization, and Machine Learning.* Addison-Wesley, 1989.

[22] J. M. Hammersley and D. C. Handscomb. *Monte Carlo Methods.* Chapman and Hall, London, 1979.

[23] R. M. Haralick, K. Shanmugan, and I. Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man and Cybernetics.* SMC-3:610–621, November 1973.

[24] R. M. Haralick and L. T. Watson. A facet model for image data. *Computer Graphics and Image Processing*, 15:113–129, 1981.

[25] C. Harris and B. Buxton. Evolving edge detectors with genetic programming. In *Proceedings of the 1996 Genetic Programming Conference*, 1996.

[26] D. C. He, L. Wang, and J. Guibert. Texture features extraction. *Pattern Recognition Letters*, pages 269–273, 1987.

[27] D. H. He and L. Wang. Texture features based on texture spectrum. *Pattern Recognition*, 24(5):391–399, 1991.

[28] J. H. Holland. *Adaptation In Natural And Artificial Systems.* The University of Michigan Press, Ann Arbor, 1975.

[29] P. V. C. Hough. Methods and means for recognizing complex patterns. *US Patent 3 069 654*, 1962.

[30] R. Jain, R. Kastuir, and B. G. Schnuck. *Machine Vision*. McGraw-Hill, 1995.

[31] E. D. Jansing. Global optimization in two dimensions after Kushner's method. Master's thesis, University of Louisville, 1992.

[32] E. D. Jansing, B. S. Allen, and D. L. Chenoweth. Edge enhancement using the fractal error metric. In *Proceedings of the 1st International Conference on Engineering Design and Automation*, 1997.

[33] E. D. Jansing, D. L. Chenoweth, and J. Knecht. Feature detection in synthetic aperture radar images using fractal error. In *Proceedings of the IEEE Aerospace Conference*, volume 1, pages 187–195, 1997.

[34] B. Julesz. Visual pattern discrimination. *IRE Transactions on Information Theory*, IT-8:84–92, 1962.

[35] J. N. Kapur, P. K. Sahoo, and A. K. C. Wong. A new method for gray level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics and Image Processing*, 29:273–285, 1985.

[36] B. H. Kaye. *A Random Walk Through Fractal Dimensions*. VCH, New York, 1989.

[37] J. M. Keller, S. Chen, and R. M. Crownover. Texture description and segmentation through fractal geometry. *Computer Vision, Graphics and Image Processing*, 45:150–166, 1989.

[38] P. A. Kelly, H. Derin, and K. D. Hartt. Adaptive segmentation of speckled images using a hierarchical random field model. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 36:1628–1641, October 1988.

[39] J. R. Koza. *Genetic Programming: On The Programming Of Computers By Means Of Natural Selection*. The MIT Press, Cambridge, MA, 1992.

[40] J. R. Koza, F. H. Bennett III, D. Andre, and M. A. Keane. Four problems for which a computer program evolved by genetic programming is competitive with human performance. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, 1996.

[41] B. B. Mandelbrot. *Fractals: Form, Chance and Dimension*. W. H. Freeman and Co., San Francisco, 1977.

[42] B. B. Mandelbrot. *The Fractal Geometry of Nature*. Freeman, San Francisco, CA, 1983.

[43] D. Marr and E. Hildreth. Theory of edge detection. In *Proceedings of the Royal Society*, volume B-207, pages 197–217, 1980.

[44] B. Moghaddam, K. J. Hintz, and C. V. Stewart. Fractal image compression and texture analysis. In *SPIE 19th AIPR Workshop*, October 1990.

[45] B. Moghaddam, K. J. Hintz, and C. V. Stewart. Dimension and lacunarity measurements of IR images using Hilbert scanning. In *SPIE International Symp. on Optical Engineering and Photonics in Aerospace Sensing*, April 1991.

[46] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, New York, second edition, 1984.

[47] H.-O. Peitgen, H. Jurgens, and D. Saupe. *Chaos and Fractals: New Frontiers of Science*. Springer-Verlag, New York, 1992.

[48] S. Peleg, J. Naor, R. Hartley, and D. Avnir. Multiple resolution texture analysis and classification. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, PAMI-6(4):518–523, 1984.

[49] A. P. Pentland. Fractal-based description of natural scenes. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, PAMI-6(6):661–674, 1984.

[50] J. C. Potts, T. D. Giddens, and S. B. Yadav. The development and evaluation of an improved genetic algorithm based on migration and artificial selection. *IEEE Transactions on Systems, Man and Cybernetics*, 24(1):73–86, January 1994.

[51] J. Prewitt. Object enhancement and extraction. In B. Lipkin and A. Rosenfeld, editors, *Picture Processing and Psychopictorics*, pages 75–149. Academic Press, 1970.

[52] E. Rignot and R. Chellappa. Segmentation of polarimetric synthetic aperture radar data. *IEEE Transactions on Image Processing*, 1(3):281–300, July 1992.

[53] G. W. Rogers, J. L. Solka, and C. E. Priebe. A PDP approach to localized fractal dimension computation with segmentation boundaries. *Simulation*, 65(1):25–36, 1995.

[54] P. K. Sahoo, D. W. Slaaf, and T. A. Albert. Threshold selection using a minimal histogram entropy difference. *Optical Engineering*, 36(7):1976–1981, July 1997.

[55] P. K. Sahoo, S. Soltani, A. K. C. Wong, and Y. C. Chen. A survey of thresholding techniques. *Computer Vision, Graphics and Image Processing*, 41:233–260, 1988.

[56] J. E. Selvage, D. L. Chenoweth, and V. Edward Gold. Geometric feature extraction using the chord transform. In *IEEE Aerospace Conference*, pages 399–405, February 1996.

[57] A. H. S. Solberg, T. Taxt, and A. K. Jain. A Markov random field model for classification of multisource satellite imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 34(1):100–113, January 1996.

[58] J. L. Solka, C. E. Priebe, and G. W. Rogers. An initial assessment of discriminant surface complexity for power law features. *Simulation*, 52(5):311–318, 1992.

[59] M. C. Stein. Fractal image models and object detection. In *Proceedings of the SPIE: Visual Communications and Image Processing II*, volume 845, pages 293–300, 1987.

[60] C. V. Stewart, B. Moghaddam, K. J. Hintz, and L. M. Novak. Fractional Brownian motion models for synthetic aperture radar imagery scene segmentation. *Proceedings of the IEEE*, 81(10):1511–1521, October 1993.

[61] A. K. C. Wong and P. K. Sahoo. A gray-level threshold selection method based on maximum entropy principle. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-19:866–871, 1989.

VITA

Eric David Jansing received both his B.S.E.S. degree in 1991 and his M.Eng. in Electrical Engineering in 1992 from the University of Louisville. His areas of interest include image processing and computer vision, evolutionary computation, and control systems. He has been an instructor in the Department of Electrical Engineering during his doctoral education at the University of Louisville. He is a member of the Institute of Electronics and Electrical Engineers (IEEE), serving as the Chair for the University of Louisville student branch during the 1995-1996 academic year. He is also a member of Eta Kappa Nu. He has received the IEEE's Gordon Northrup award in 1996 and has been elected into the *Who's Who Among Students In American Universities and Colleges.*

Appendix B – Publications Supported By ONR Grant N00014-92-J-4096
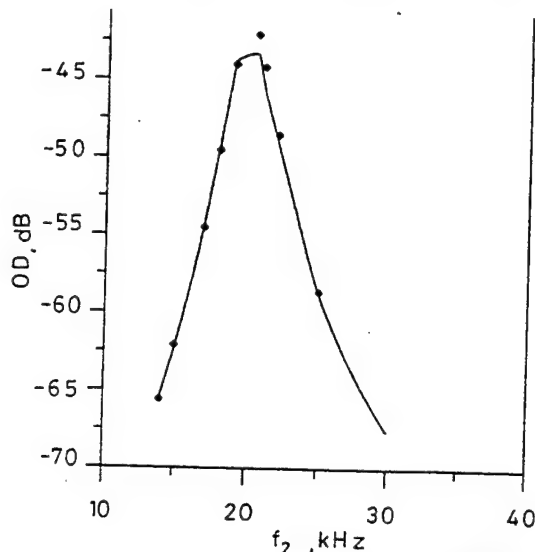
Fig. 3 *Nonlinear distortion for $V_{i1} = 0.2V$, $V_{i2} = 0.2V$ and $F_1 = 19.8kHz$*

——— predicted
◆ measured

X. Zeng, P. Bowron and A. A. Muhieddine (*Department of Electronic and Electrical Engineering, University of Bradford, West Yorkshire, BD7 1DP, United Kingdom*)

X. Zeng: On leave from University of Electronic Science and Technology of China, Chengdu 610054, People's Republic of China

References

1 BOWRON, P., MOTLAGH, M.R.J., and MUHIEDDINE, A.A.: 'Harmonic characterisation of feedback systems incorporating saturation nonlinearities', *Electron. Lett.*, 1991, **27**, (20), pp. 1865–1867

2 BUONOMO, A.: 'Time-domain analysis of nonlinear circuits with periodic excitation', *Electron. Lett.*, 1991, **27**, (1), pp. 65–66

3 BORYS, A.: 'On intermodulation and harmonic distortion in single-amplifier active filters', *J. Audio Eng. Soc.*, 1980, **28**, (10), pp. 706–712

4 ZENG, X., BOWRON, P., and MUHIEDDINE, A.A.: 'HD and IMD prediction techniques for active filters'. Proc. of ISCAS'94, May 30–June 2, 1994, London

5 ZENG, X.: 'Frequency-domain-continuation inter-modulation balance method for the analysis of nonlinear microwave circuits with multifrequency excitations', *IEE Proc. H*, 1993, **40**, (4), pp. 259–262

6 HAYES, J.G.: 'Numerical approximation to functions and data' (Athlone Press, London, 1970)

Fractal error for detecting man-made features in aerial images

B.E. Cooper, D.L. Chenoweth and J.E. Selvage

Indexing terms: Remote sensing, Image processing, Fractals

A technique is proposed for aiding photointerpreters in detecting man-made features in aerial reconnaissance images. The technique, which uses a metric called fractal error, is based on the observed propensity of natural image features to fit a fractional Brownian motion model. Man-made features usually do not fit this model well, and consequently the fractal error metric may be used as a discriminant function for detecting man-made scene features.

Introduction: Analysis and interpretation of aerial rec images often require the analyst to view large amou data for the purpose of detecting man-made features. ter a new photointerpretation metric, referred to as fra applied to the problem of detecting man-made featu reconnaissance images. It is based on the observation features in high-resolution grey shade aerial images tional Brownian motion (fBm) mathematical model. 7 text of Mandelbrot [1] links fractal geometry to the and infinite detail unique to natural features, and Pentl tends to support this hypothesis.

By computing the fractal error locally over an entir possible to accentuate local scene features that have origins. This information can then be used by the phot for a variety of purposes, such as change analysis and ture extraction.

Fractal error: We may describe fractional Brownian terms of the properties of the signal's increments. The must be normally distributed with a mean of zero and proportional to the $2H$ power of the Euclidean distan the locations where any two signal measurements are more common form shown below combines these two into a single expression in terms of the mean of the m the increments. In this case the signal of interest is a t sional grey shade aerial image.

Let $G(\mathbf{x})$ refer to a two-dimensional grey shade im will be compared to the fBm signal model. Thus, $\mathbf{x} = (x$ ifies the discrete row and column co-ordinates of t Assume that the discrete valued function $G(\mathbf{x}) \in [0, ...$ sents the grey shade at location $\mathbf{x}$. If $G(\mathbf{x})$ fits the fBm m the following equations hold for some H (where $0 < H$ some k (where $k > 0$):

$$E[|G(\mathbf{x}_2) - G(\mathbf{x}_1)|] = k|\mathbf{x}_2 - \mathbf{x}_1|^H$$

$$E[|\Delta G_{|\Delta \mathbf{x}|}|] = k|\Delta \mathbf{x}|^H$$

The fractal dimension D is related to the parameter H b $1 - H$, where E is the number of independent variables nal. Here, $E = 2$.

The distance between a pair of pixels is $|\Delta \mathbf{x}| = |\mathbf{x}_2 - \mathbf{x}_1$ increment, or difference in grey shades of the two pixels. $= G(\mathbf{x}_2) - G(\mathbf{x}_1)$. The average absolute grey shade differe across several pairs of pixels with spacings of the same m will follow the above exponential scaling if the image fit model. Moreover, a linear equation may be obtained by t logarithm of this equation:

$$\ln(E[|\Delta G_{|\Delta \mathbf{x}|}|]) = \ln(k) + H \ln(|\Delta \mathbf{x}|)$$

Using least-squares linear regression, the estimated values may be determined from a collection of pixel distances associated average absolute changes in grey shade. The can be obtained locally using a window operator that is si compatible with the resolution of the image and its featur

The error at a particular pixel spacing within the win be defined as the difference between the actual and estim ues as follows:

$$error_{|\Delta \mathbf{x}|} = E[|\Delta G_{|\Delta \mathbf{x}|}|] - \bar{k}|\Delta \mathbf{x}|^H$$

Finally, we may use the root mean square (RMS) of these ual errors within the window to characterise the overall lo in the estimate. Letting n be the number of pixel distances ered in the window.

$$RMS\ error = \sqrt{\frac{1}{n} \sum_{|\Delta \mathbf{x}|} (error_{|\Delta \mathbf{x}|})^2}$$

The RMS error measures the degree to which the observe fits the fBm model in a window centred at the pixel of and it is referred to here as the fractal error. Equivale complement of the fractal error may be considered as a of the 'fractalness' of the image at the pixel of interest.

In this study the fractal error is implemented using wind of 5×5, 7×7 and 9×9 pixels. Rather than using every possit spacing within the window, computational effort is redu examining only those distances from the central pixel of t window. For example, Fig. 1 illustrates the five differen
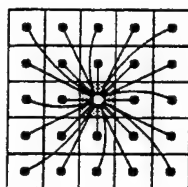
Fig. 1 *Pixel spacing used with 5 × 5 fractal error*

$\Delta x = 1$
$\Delta x = \sqrt{2}$
$\Delta x = 2$
$\Delta x = \sqrt{5}$
$\Delta x = \sqrt{8}$

spacings used within a 5×5 window. The 7×7 and 9×9 operators are similar to the 5×5 window.
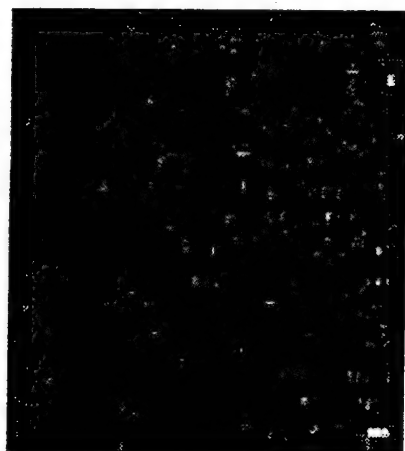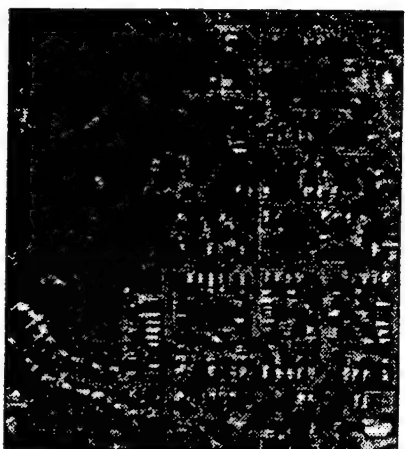
Fig. 2 *Original images*

a April
b October

Fig. 3 *Binary fractal error images*

a April
b October

nal image and the threshold fractal error images that the fractal error performs reasonably well in detecting man-made features. It is also fairly insensitive to changes in illumination, contrast and season. Note the brighter lighting and greater contrast in the October image of Fig. 2 compared to the April image, as well as the distinct increase of foliage in the October scene.

Conclusions: It has been demonstrated that man-made scene features can be detected in grey shade aerial reconnaissance images using the fractal error metric. The metric provides the photointerpreter analyst with a tool for automatically processing large image databases to determine the presence of, and estimate the extent of, man-made features in grey shade reconnaissance imagery.

17 January 1994

B.E. Cooper, D.L. Chenoweth and J.E. Selvage (*Computer Science and Engineering Program, University of Louisville, Louisville, KY 40292, USA*)

References

1   MANDELEBROT, B.B.: 'The fractal geometry of nature' (W.H. Freeman and Co., New York, 1983)
2   PENTLAND, A.P.: 'Fractal-based description of natural scenes', *IEEE Trans.*, 1984, **PAMI-6**, (6), pp. 661–674

Results: To illustrate the technique a pair of images will be considered to demonstrate qualitatively the ability of the fractal error operator to detect man-made image features. The images consist of aerial photographs of a residential area with a fairly high content of deciduous vegetation. The same scene was photographed in April and in October. This was done to evaluate the robustness of the metric relative to seasonal variations and changes in contrast and illumination that occurred at the different times. The primary man-made objects in the test images are about 50 to 60 pixels in size. The pair of images is shown in Fig. 2. Fig. 3 illustrates the fractal error of each of these images, using the 7×7 window size, thresholded so that features with large fractal errors appear black against a white background. It is obvious by comparing the origi-

a study of the mathematical concerns of various algorithms to estimate the fractal dimension, Pruess confirms this sentiment, stating that one of the essential attributes of an estimation algorithm is to "provide some statistical information as to the accuracy and sensitivity of the output." [13, p. 10] Before estimating the fractal characteristics of an object, it is important to determine if such a model is appropriate. Furthermore, the degree to which an object fits the fractal model is an important characteristic itself.

Fractional Brownian Motion

The predominant measure of a fractal is dimension. Therefore, the error in the estimation of the fractal dimension was examined, although the error in the estimation of other fractal characteristics could (and should) be considered. Following the historical precedent of Mandelbrot [8], Pentland [12], and others, the fractal dimension used here is based upon the H parameter of fractional Brownian motion.

Fractional Brownian motion may be described by the properties of the signal's increments, which must be normally distributed with a mean of zero and a variance proportional to the $2H$ power of the Euclidean distance between the two measurements. Let $B_H(\mathbf{t})$ represent a fBm signal, where $\mathbf{t}$ is a vector containing E independent variables. Then the properties of the increments $\Delta B_H = B_H(\mathbf{t}_2) - B_H(\mathbf{t}_1)$ are expressed below, where k represents the proportionality constant of the variance and H is a parameter between zero and one, exclusively ($0 < H < 1$):

$$E[\ B_H(\mathbf{t}_2) - B_H(\mathbf{t}_1)\ ] = 0 \tag{1}$$

$$\mathrm{var}[\ B_H(\mathbf{t}_2) - B_H(\mathbf{t}_1)\ ] = \sigma^2 |\mathbf{t}_2 - \mathbf{t}_1|^{2H} \tag{2}$$

The fractal dimension D is related to the parameter H by the formula $D = E + 1 - H$, where E is the topological dimension (equivalently, the number of independent variables of the signal).*

Since $\mathrm{var}[Y] = E[Y^2] - E[Y]^2$, these two equations may be combined to give the expression

$$E[\ (B_H(\mathbf{t}_2) - B_H(\mathbf{t}_1))^2\ ] = \sigma^2 |\mathbf{t}_2 - \mathbf{t}_1|^{2H} \tag{3}$$

However, the more common form is below, which may be derived through a transformation of random variables, as shown in the Appendix. (The constant k is proportional to the standard deviation σ.)

$$E[\ |B_H(\mathbf{t}_2) - B_H(\mathbf{t}_1)|\ ] = k\ |\mathbf{t}_2 - \mathbf{t}_1|^{H} \tag{4}$$

When considering a two-dimensional signal, such as an image, there will be two independent variables ($E = 2$). The notation $G(\mathbf{x})$ refers to such a signal, which will be compared to the fBm signal $B_H(\mathbf{t})$. Thus, $\mathbf{x} = (x_r, x_c)$, which specifies the discrete row and column coordinates. The function $G(\mathbf{x}) \in [0, ..., 255]$ is a discrete value representing the gray shade at location $\mathbf{x}$.

If $G(\mathbf{x})$ is assumed to be fBm, then the following equations hold for some H ($0 < H < 1$) and k ($k > 0$):

* Mandelbrot mentions this relationship as early as [8], but does not elaborate upon it. See Voss [17] for a lucid explanation.

279

formation of galactic clusters. In contrast to traditional geometry, fractal geometry readily accommodates the complexity and infinite detail present in nature.

Modeling Natural Terrains with Fractals

Mandelbrot [8] noticed that natural terrain surfaces may be modeled by two-dimensional fractional Brownian motion with startling effectiveness. Introduced by Mandelbrot and Van Ness [10], fractional Brownian motion (sometimes abbreviated *fBm*) is a generalization of Brownian motion, which describes the complex, erratic movement of a particle, subjected to the collisions of the other particles in its surrounding medium. Voss [16] established a link between the spectral distribution of fBm and the fractal dimension. Based on this, he used Fourier synthesis to generate natural terrains and even complete landscapes. Pentland [12] argued that natural terrain should follow a fractal model, since it is formed by fractal processes such as aggregation and erosion. Furthermore, he asserted that an image intensity surface of a fBm surface will also produce fBm, under normal lighting conditions. In other words, not only may terrain be modeled by fBm, but images of such terrain also.

Modeling Natural Image Textures

Recognizing that fBm may be used to analyze images of natural terrain, Pentland [12] extended fBm to image textures in general. He found that fBm relief maps of varying fractal dimension correlated very closely with the human perceptual notion of roughness, a fundamental characteristic of texture. Estimating the fractal dimension from the Fourier transforms of local neighborhoods, Pentland distinguished the textures within a sampling of Brodatz textures and textures from natural scenes. The estimated fractal dimension was also used to segment the regions within images of a desert, San Francisco Bay and Mount Dawn. Keller et al. [6] developed a one-dimensional fBm model to distinguish between silhouettes of trees and mountains.

Natural image texture models are not restricted to using only the fractal dimension. In later work, Keller et al. [5] supplemented the fractal dimension with a lacunarity estimate to segment a set of Brodatz textures and a scene of trees. Likewise, Vehel [15] implemented an alternate form of each lacunarity and the fractal dimension to classify images of human lungs. Generalizing the normally scalar fractal dimension, Kaneko [4] used a 2×2 fractal dimension matrix to capture directional dependencies. The eigenvalue of this matrix was used to characterize Brodatz textures. Dennis and Dessipris [2] studied the derivative (rate of change) of the fractal dimension as a texture signature across a range of pixel distances.

THE MODEL

Motivation

In spite of the variety of the fractal measures discussed above, all of them (except the rely upon one fundamental assumption: that the image texture is a fractal. Presumably, this situ results from Pentland's original assertion that images of natural terrain may be modeled by (specifically fBm) approach [12]. However, Pentland assumes that the scene's reflectanc uniform. When this condition does not hold, the validity of the fractal model is questio

Image Segmentation Using Modified Neural Network Techniques

V. Edward Gold, Darrel L. Chenoweth, and John E. Selvage

Electrical Engineering Department
University of Louisville
Louisville, KY 40292

ABSTRACT

This paper describes an application of neural networks in segmenting gray shade images. It describes a method for ranking pixel features relative to their ability to discriminate among different image segment classes. A neural classifier is proposed which operates on pixel feature vectors as inputs to the network, each feature having a variable weight. The weights are iteratively changed to obtain dense and highly separated clusters. The resulting weights are indicative of the usefulness, or rank, of the features.

Key Words: Image segmentation, feature ranking, neural classification, Kohonen maps

1. INTRODUCTION

This paper describes a unique implementation of neural network technology for image segmentation applications. This algorithm characterizes each pixel of an image as a vector. A k-means classifier (implemented as a Kohonen[1] network) then assigns the pixel to an image segment class. The results are then output as a segmented image.

Many neural network approaches rely on a classifier to determine the relationship between the input vectors and the desired outputs. Assuming the network achieves acceptable performance, the designer is still left with little or no understanding of what the network found valuable in the input data. Thus, further optimization may be difficult or impossible. To greatly reduce the number of input variables, we propose a feature extraction stage to remap the input pixels into feature vectors. The advantages and the disadvantages of this approach are discussed in this paper.

The neural network designed for this implementation is not a neural network in the traditional sense of the term. The neural network is generated after the training data has been fully clustered and analyzed. Thus, the neural network does not come into existence until training is completed. The design technique for this implementation will be discussed in the sections that follow.

---

2. FUNCTIONALITY

Image processing applications often begin with a need to reduce the data in an input image into a significantly smaller amount of information. For example, in fingerprint imagery, the inter-relationships between the ridge events, endings and bifurcations yield all the information necessary to compare or identify the print. A fingerprint examiner is not concerned with how dark the ridges appear, nor is he concerned with the rotation, scale, edge magnitudes, or any other information contained in the image. In target tracking applications, as another example, the image processing algorithms must remove all of the unwanted information and extract only information about the target's position and possibly its signature.

Frequently, the process of reducing an image begins with the segmentation of the image into regions of similar compositon. This paper will discuss a technique for performing this segmentation using a pixel-based classifier implemented via a Kohonen neural network architecture.

Before the neural network can operate on the pixel data, features are extracted from the pixel and its neighbors. These features attempt to fully describe the pixel in every way that might help distinguish one class of pixels from all other classes. For example, in a satellite photo, pixel classes such as "tree", "building", "road", and "ocean" are applicable. Finger print image classes might be "ridge", "valley", and "background". Features would then be any manipulation of the input data that reveals some information about the pixel of interest. These can be based on mathematical analysis or on heuristics that make sense to the algorithmist.

The algorithm implemented for this paper considers 15 heuristic texture and gray shade features to be tested in the pixel classifier. One of the main goals in designing this algorithm was to determine a mechanism for evaluating how well each feature discriminates between the different classes. Since feature salience depends heavily on the imagery and its contents, a wide variety of features are required for a robust algorithm. This also requires that the algorithm make determinations as to the relative discrimination capability of each feature. Assuming it can perform this task, the algorithmist would then optimize his design for a specific application by retaining only the salient features.

The features will be normalized before the clustering algorithm attempts to organize them. The clustering process measures similarity between input vectors by the Euclidean distance function. If one feature was scaled by a large constant, then the classifier would assign significantly more weight to variations in that feature. Before the relative discriminating capabilities of each feature are known, no such bias should exist.

The clustering algorithm involves an iterative process of proposing a cluster organization, testing that organization, tuning the feature weighting coefficients, and adjusting the operational radius for each cluster. The classifier itself consists of a single layer of operators or nodes. During run time operation, the classifier evaluates each input sample using a weighted Euclidean distance

metric. Before discussing the training process, a short discussion of the run time operation is appropriate.

To determine the input sample's class, the classifier defines several clusters, or distributions, in feature space. Each of the previously mentioned nodes represents one of these clusters. This is done so that we can represent data that may be multi-modal more accurately. To explain the rationale for this, imagine a fruit classifier that is trained on apples, oranges and grapes. The object features might be color, shape, and size. If a wide array of apple samples are selected, the color feature would likely offer little help in discriminating apples from the other fruits because green apples were treated as part of the same distribution as the red and yellow apples.

Figure 1 illustrates the fruit classifier sample distribution with size and color features. Clearly, size alone will not provide enough information to discriminate between apples and oranges. The middle figure identifies a potential interpretation of the input samples into meaningful distributions. A classifier that interprets the samples in this manner should be able to accurately classify most test samples correctly. The lower figure shows a poor interpretation of the sample distribution. A classifier that assumes these distributions will probably not perform with acceptable accuracy.

An obvious and easy remedy would be to create classes such as red_apple, green_apple, etc. A better approach would employ clustering techniques to accommodate these intra-class differences, and account for them as they occur. This is very important in image processing applications where the designer might not fully understand the nature of the sample distributions. The self-organizing clustering algorithm that we propose enables the classifier to operate autonomously on any feature vector samples without customization to the specific application. This capability is especially useful in systems intended for novice operators with no knowledge of probability.

The training procedure essentially attempts to determine a minimal data representation of the input sample feature distribution. The weighted distance to each node center is computed and that distance is then checked against the node's maximum acceptable distance. If the distance is within the node's "sphere of operation", the node continues to compete for the sample. The closest node with acceptable distance to the sample is then chosen as the node that will determine the class of the sample. Upon completion of training, the inverse covariance matrix is computed for each class of each node. This is used to compute the Mahalanobis distance from each class mean to the sample currently being classified. The classifier determines a sample's class by first locating the closest node, then by applying the Mahalanobis distance formula for each class within the node. The minimum distance determines the selected class.

Feature Selection and Evaluation

The features selected for use in the algorithm discussed in this paper are listed in Table 1. Each of these features reveals something about image texture or gray level in the region centered around the pixel of interest. Additional features such as the difference between the pixel intensity and its

$$E[\ |G(\mathbf{x}_2) - G(\mathbf{x}_1)|\ ] = k\ |\mathbf{x}_2 - \mathbf{x}_1|^H \tag{5}$$

$$E[\ |\varDelta G_{|\varDelta\mathbf{x}|}|\ ] = k\ |\varDelta\mathbf{x}|^H \tag{6}$$

The distance between a pair of pixels is $|\varDelta\mathbf{x}| = |\mathbf{x}_2 - \mathbf{x}_1|$, and the difference in gray shade between these pixels is $\varDelta G_{|\varDelta\mathbf{x}|} = G(\mathbf{x}_2) - G(\mathbf{x}_1)$. The average absolute gray shade difference, taken across several pixel spacings of the same magnitude, will follow the above exponential scaling. A linear equation may be obtained by taking the logarithm of each side:

$$\ln(E[\ |\varDelta G_{|\varDelta\mathbf{x}|}|\ ]) = \ln(k) + H\ln(|\varDelta\mathbf{x}|) \tag{7}$$

Using least squares linear regression, the estimated values $\overline{H}$ and $\overline{k}$ may be determined from a collection of pixel distances and their associated average absolute changes in gray shade.

Fractal Error

However, $G(\mathbf{x})$ may or may not fit the fBm model. Although the above method will compute estimates for H and k, the validity of these values depends upon how well $G(\mathbf{x})$ corresponds to fBm. Thus, the error in the estimation should be considered. The error at a particular pixel spacing is

$$\text{error}_{|\varDelta\mathbf{x}|} = E[\ |\varDelta G_{|\varDelta\mathbf{x}|}|\ ] - \overline{k}\ |\varDelta\mathbf{x}|^{\overline{H}} \tag{8}$$

The root mean square (RMS) of these individual errors characterizes the overall error in the estimation, shown below. (Let n be the number of pixel distances considered.)

$$\text{RMS error} = \sqrt{\frac{1}{n}\sum_{|\varDelta\mathbf{x}|}(\text{error}_{|\varDelta\mathbf{x}|})^2} \tag{9}$$

The RMS error measures the degree to which the observed signal fits the fBm model and is referred to here as the *fractal error*. Equivalently, the complement of the fractal error may be considered as a measure of the "fractalness" of the signal.

EXAMPLE IMPLEMENTATIONS AND RESULTS

Developing a Local Measure

The development of a fractal error feature was oriented toward textures in aerial gray shade reflectance images containing features such as trees, grass, shadows, water, soil and buildings. Because the regions in these images may be small, a small neighborhood of pixels was desired. Window sizes of 5×5, 9×9, 13×13 and 17×17 were studied, but the smallest size, 5×5, most suitable in the examples which are discussed in this paper. Also, this size offered greater s making it more practical to compute the fractal error within each pixel's neighborhood.

Another factor was the choice of which pixel spacings to consider in computing th error. For a window of size $w \times w$, there are $(w^4 - w^2)/2$ possible spacings. To reduce th of computations, two approaches were examined. By considering only those pixel sp horizontal and vertical directions, only $w^3 - w^2$ data points must be used. A furthe

sults from a second technique. It examines all distances and angles within the window, but only with respect to the center pixel. There are $w^2 - 1$ such pixel spacings. Figure 1 illustrates the pixel spacings used within a 5×5 window for the center-oriented method. Intuitively, this approach seems to localize the measure more by emphasizing the center pixel, plus it strives to make the measure more directional independent than the former approach.

Applying Fractal Error to Standard Brodatz Textures

Brodatz textures [1] are often used as standards for evaluating texture-based algorithms. A test image was developed from three of these textures to investigate the discriminatory capabilities of the localized fractal error operator. The fractal error operator was applied to a collage of Brodatz textures in Figure 2. (The top of Figure 2a is "paper," the bottom is "cork" and the center is "sand.") A reasonable degree of separation is evident in Figure 2b, and the three different textures are are clearly discernable.

Segmenting Aerial Imagery

The development of a fractal error feature was oriented toward the segmentation of aerial reconnaisance images, into categories such as trees, grass, soil, water, shadows, roads and buildings. For example, the regions of interest in the shore scene (Figure 3a) might be water, foliage (trees)
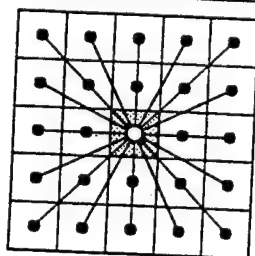
Pixel distances

$\Delta x = 1$
$\Delta x = \sqrt{2}$
$\Delta x = 2$
$\Delta x = \sqrt{5}$
$\Delta x = \sqrt{8}$

Figure 1   Pixel Spacings Used with the Center-Oriented Method
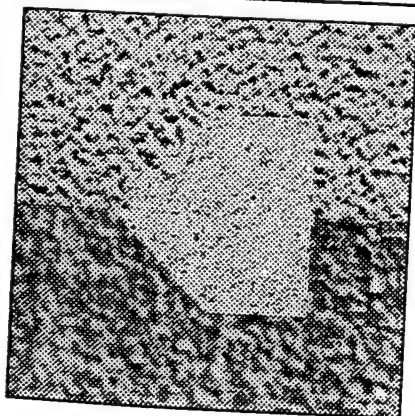
(a) Original Image
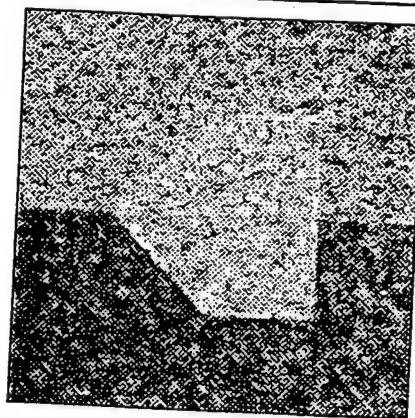
(b) Fractal Error Image

Figure 2   Fractal Error Applied to a Collage of Brodatz Textures

(a) Original Image *(b) Fractal Error Image, Negated* *(c) Segmented Image*

Figure 3 *Fractal Error Applied to the Shore Scene*
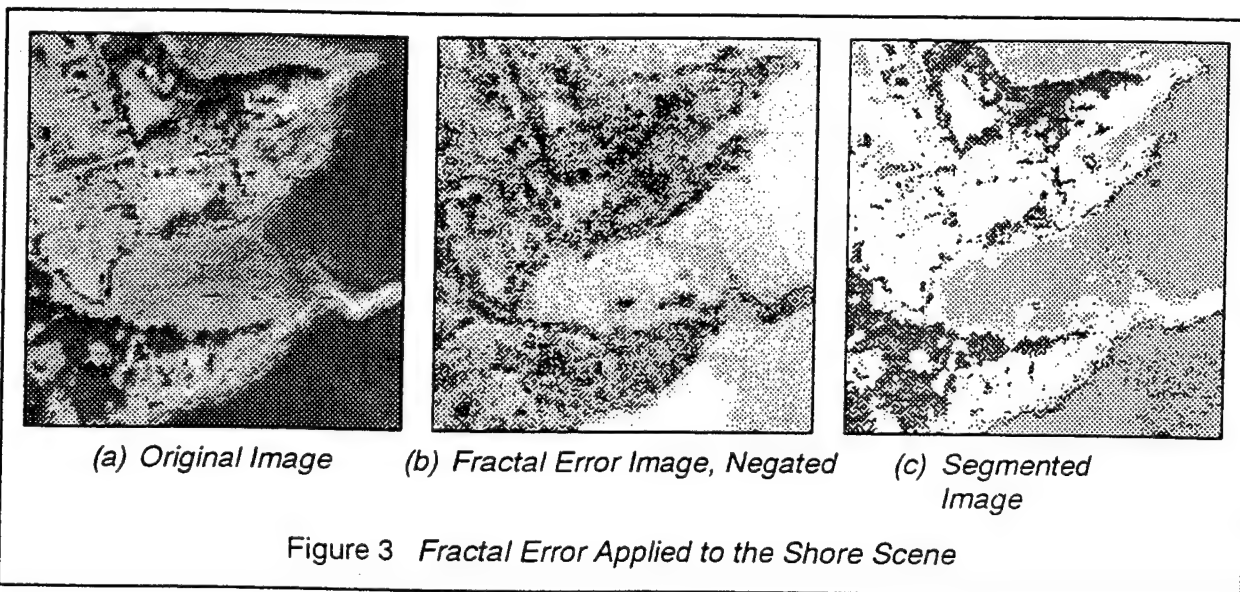
and non–foliated land. Although gray shade is often the predominant feature for distinguishing re-
gions in aerial images, gray shade alone may be inadequate. Texture features are necessary, and be-
cause of the wide variety of textures present and the frequent occurrence of small regions a texture
feature for aerial imagery must be somewhat general–purpose and defined on a small neighborhood.

Fractal Error as a Texture Feature – An Aerial Image Segmentation Application

Figure 3a illustrates a shore scene (278 x 269) containing water, foliage and non–foliated
land. Clearly, the gray shade of the water overlaps that of both the foliage and some of the open
fields. The fractal error image in Figure 3b, negated to improve visibility, demonstrates that the wa-
ter tends to have a low fractal error (light gray shade). The non–foliated land has a greater error,
and the foliage has a medium to high error. Thus, a combination of gray shade and fractal error may
be used to classify the pixels in this image, whereas either one of the features individually will fail.
A supervised maximum likelihood classifier was designed to segment the image, on a pixel by pixel
basis, using gray shade and local fractal error as the pixel features. The segmentation result is shown
in Figure 3c, separating foliage (dark gray), water (light gray) and non–foliated land (white).

A farm land scene (512 x 512) is shown in Figure 4a. The light shaded soil areas are clearly
distinguishable solely on the basis of gray shade intensity. However, the grassy fields and foliage
require a texture measure to be separated. Figure 4b shows the negated fractal error image, in which
the foliage tends to have a larger error (darker gray shade) than the fields. In Figure 4c, the image
has been classified as foliage (dark gray), grass (light gray) or soil (white).

282
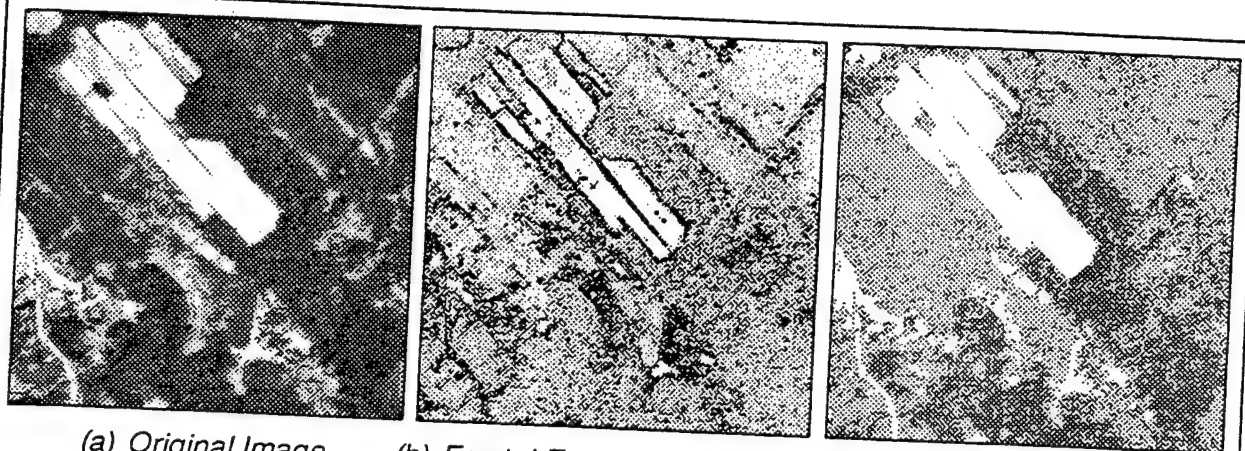
(a) Original Image   *(b) Fractal Error Image, Negated*   *(c) Segmented Image*

Figure 4   *Fractal Error Applied to the Farm Scene*

Finally, Figure 5a (512 x 512) introduces a mixture of natural and cultural (e.g., roads, build-
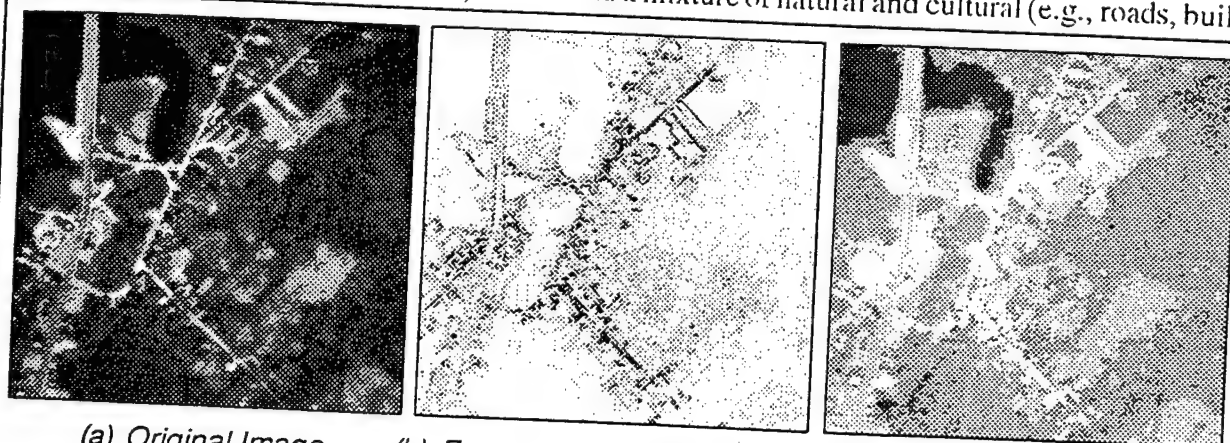
(a) Original Image   *(b) Fractal Error Image, Negated*   *(c) Segmented Image*

Figure 5   *Fractal Error Applied to the Mixed Scene*

ings) content. The lake in the upper left, the trees, the fields and cultural objects are generally separated in the negated fractal error image, shown in Figure 5b. The segmentation in Figure 5c identifies four classes, from dark to light: water, foliage, open fields and cultural objects.

Ground truth for these three images was provided by a trained scene analyst, and the overall classification accuracy was found to be between 85 and 90 percent.

CONCLUSIONS

Few researchers have considered the error in estimating fractal characteristics. In this investigation we have attempted to address the issue and have defined a new texture descriptor referred to as *fractal error*. Defined within a much smaller neighborhood than most other fractal characteris-

283

tics, the fractal error was developed as a localized texture feature. The fractal error texture feature and the gray shade have been shown to be an effective set of features for classifying pixels in aerial images in aerial image segmentation applications.

Other Applications of Fractal Error

The fractal error associated with non-fractal image features will be relatively large. Since cultural or man-made features often have a non-fractal appearance, the fractal error metric could be used to screen cultural features in aerial reconnaissance images. To a mission planner /scene analyst this may be useful information. We are currently investigating several examples containing cultural features to quantify the effectiveness of fractal error for detecting cultural feature content in aerial imagery.

REFERENCES

[1] P. Brodatz, *A Photographic Album for Artists and Designers,* New York: Dover, 1966.

[2] T. J. Dennis and N.G. Dessipris, "Fractal Modelling in Image Texture Analysis," *IEE Proceedings,* Vol. 136, Part F, No. 5, pp. 227–235, October 1989.

[3] K. Falconer, *Fractal Geometry - Mathematical Foundations and Applications,* New York: John Wiley and Sons, Ltd., 1990.

[4] H. Kaneko, "A Generalized Fractal Dimension and Its Application to Texture Analysis," *IEEE International Conference on Acoustics, Speech, and Signal Processing,* pp. 1711–1714, 23–26 May 1989.

[5] J. M. Keller, S. Chen and R. M. Crownover, "Texture Description and Segmentation through Fractal Geometry," *Computer Vision, Graphics, and Image Processing,* Vol. 45, pp. 150–166, 1989.

[6] J. M. Keller, R. M. Crownover and R.Y. Chen, "Characteristics of Natural Scenes Related to the Fractal Dimension," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. PAMI-9, No. 5, pp. 621–627, September 1987.

[7] K. I. Laws, *Textured Image Segmentation,* Ph.D. dissertation, Department of Engineering, University of Southern California, Los Angeles, CA, 1980.

[8] B. B. Mandelbrot, "Stochastic Models for the Earth's Relief, the Shape and the Fractal Dimension of the Coastlines, and the Number–Area Rule for Islands," *Proceedings of the National Academy of Sciences USA,* Vol. 72, pp. 3825–3828, 1975.

[9] B. B. Mandelbrot, *Fractals: Form, Chance and Dimension,* San Francisco: W. H. Freeman and Co., 1977.

Input Sample Feature Distribution

Size

Color

Red   Orange   Green

Clustered Samples

Size

Mean Red Apples +

Mean Oranges +

Mean Green Apples +

Color

Red   Orange   Green

Clusters Interpreted as Unimodal Distributions

Size

Mean Apples +

Mean + Oranges

Color

Red   Orange   Green
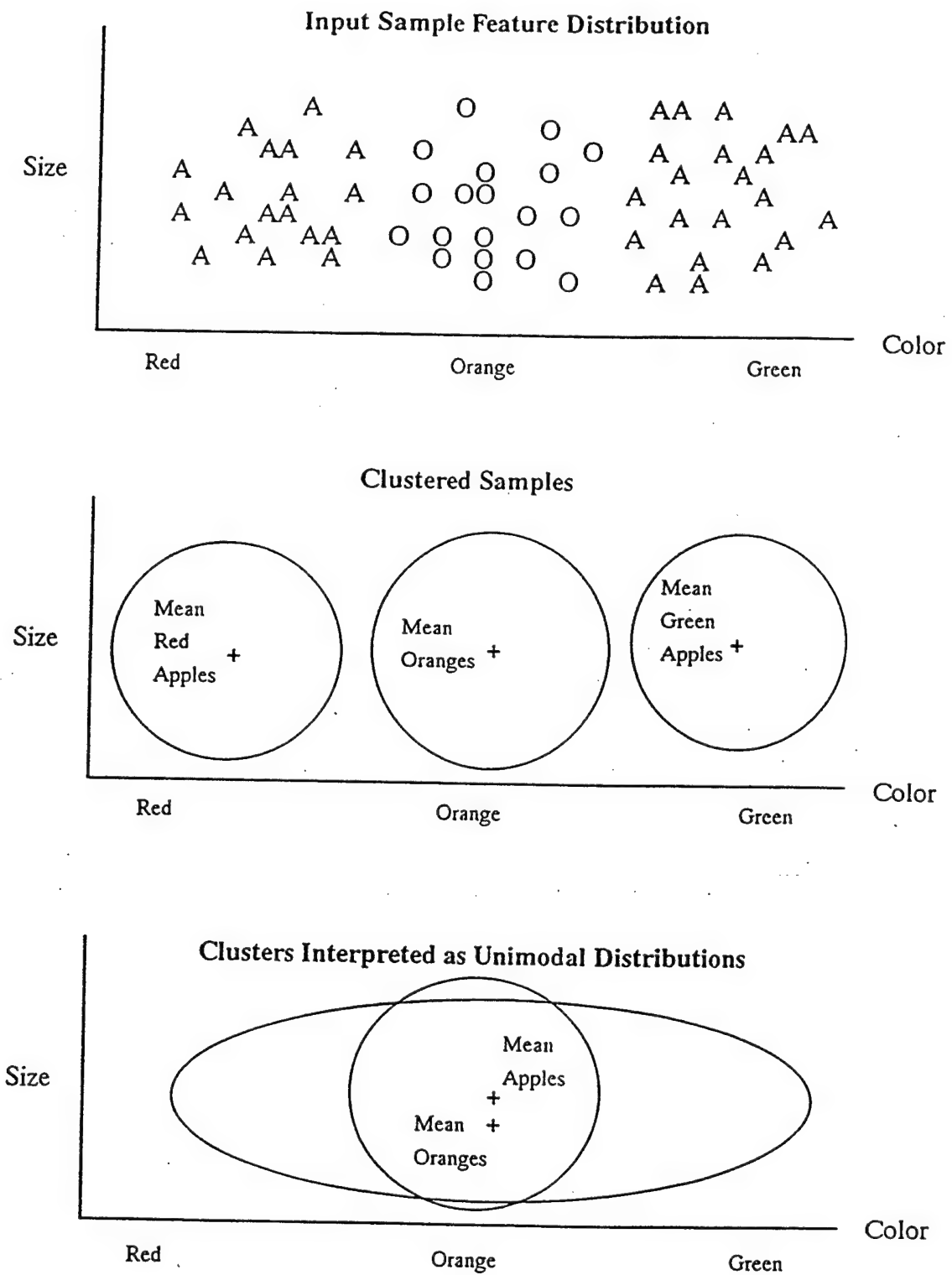
Figure 1

mean ($I_{x,y}$ – $Mean_{3x3}$) can be constructed from these features as well. This feature would likely correlate better than the pixel intensity alone since it measures offset above the average rather than intensity alone.

TABLE 1
PIXEL FEATURES TESTED IN IMAGE SEGMENTER

Feature	Definition
$Sobel\_Magnitude_X$	$(2 * P_{x+1,y} + P_{x+1,y-1} + P_{x+1,y+1}) -$ $(2 * P_{x-1,y} + P_{x-1,y-1} + P_{x-1,y+1})$
$Sobel\_Magnitude_Y$	$(2 * P_{x,y-1} + P_{x-1,y-1} + P_{x+1,y-1}) -$ $(2 * P_{x,y+1} + P_{x-1,y+1} + P_{x+1,y+1})$
$Sobel\_Magnitude$	$\sqrt{Sobel\_Mag^2_X + Sobel\_Mag^2_Y}$
Gradient	$\max$ $(abs(P_{x,y} - P_{x-1,y-1}),$ $abs(P_{x,y} - P_{x,y-1}),$ $abs(P_{x,y} - P_{x+1,y-1}),$ $abs(P_{x,y} - P_{x-1,y}),$ $abs(P_{x,y} - P_{x+1,y}),$ $abs(P_{x,y} - P_{x-1,y+1}),$ $abs(P_{x,y} - P_{x,y+1}),$ $abs(P_{x,y} - P_{x+1,y+1}))$
$Mean_{3x3}$	$1/9 \ \Sigma_{\iota=x-1,x+1}\Sigma_{j=y-1,y+1} \ P_{i,j}$
$Variance_{3x3}$	$1/9 \ \Sigma_{\iota=x-1,x+1}\Sigma_{j=y-1,y+1} \ P^2_{i,j} -$ $(1/9 \ \Sigma_{\iota=x-1,x+1}\Sigma_{j=y-1,y+1} \ P_{i,j})^2$
$Mean_{5x5}$	$1/25 \ \Sigma_{\iota=x-2,x+2}\Sigma_{j=y-2,y+2} \ P_{i,j}$
$Variance_{5x5}$	$1/25 \ \Sigma_{\iota=x-2,x+2}\Sigma_{j=y-2,y+2} \ P^2_{i,j} -$ $(1/25 \ \Sigma_{\iota=x-2,x+2}\Sigma_{j=y-2,y+2} \ P_{i,j})^2$
$Mean_{7x7}$	$1/49 \ \Sigma_{\iota=x-3,x+3}\Sigma_{j=y-3,y+3} \ P_{i,j}$
$Variance_{7x7}$	$1/49 \ \Sigma_{\iota=x-3,x+3}\Sigma_{j=y-3,y+3} \ P^2_{i,j} -$ $(1/49 \ \Sigma_{\iota=x-3,x+3}\Sigma_{j=y-3,y+3} \ P_{i,j})^2$
$Busyness_X$ [2]	$1/6$ $(abs(P_{x-1,y-1} - P_{x,y-1}) +$ $abs(P_{x-1,y} - P_{x,y}) +$ $abs(P_{x-1,y+1} - P_{x,y+1}) +$ $abs(P_{x+1,y-1} - P_{x,y-1}) +$ $abs(P_{x+1,y} - P_{x,y}) +$ $abs(P_{x+1,y+1} - P_{x,y+1}))$

TABLE 1 (Continued)

PIXEL FEATURES TESTED IN IMAGE SEGMENTER

Feature	Definition
Busyness$_Y$ [3]	$1/6$ $(abs(\ Px-1,y-1 - Px-1,y\ ) +$ $abs(\ Px,y-1 - Px,y\ ) +$ $abs(\ Px+1,y-1 - Px+1,y\ ) +$ $abs(\ Px-1,y+1 - Px-1,y\ ) +$ $abs(\ Px,y+1 - Px,y\ ) +$ $abs(\ Px+1,y+1 - Px+1,y\ )\ )$
Busyness$_{D45}$ [3]	$1/4$ $(abs(\ P_{x-1,y-1} - P_{x,y}\ ) +$ $abs(\ P_{x+1,y+1} - P_{x,y}\ ) +$ $abs(\ P_{x,y-1} - P_{x-1,y}\ ) +$ $abs(\ P_{x+1,y} - P_{x,y+1}\ )\ )$
Busyness$_{D135}$ [3]	$1/4$ $(abs(\ Px-1,y+1 - Px,y\ ) +$ $abs(\ Px+1,y-1 - Px,y\ ) +$ $abs(\ Px,y-1 - Px+1,y\ ) +$ $abs(\ Px+1,y - Px,y-1\ )\ )$
Busyness [2]	min $(Busyness_X,\ Busyness_Y,$ $Busyness_{D45},\ Busyness_{D135}\ )$

After clustering the samples, the classifier evaluates the inter-class variance and the intra-class variances for each feature. The ratio of the inter-class variance to the sum of the intra-class variances indicates a "scatter ratio". A very high inter-class variance identifies that the feature is widely spread across multiple classes. A small intra-class variance indicates the samples of a single class within this cluster are all packed into a small region. A "good" feature would then have a very high inter-class variance and all the intra-class variance terms would be small. Using these ratios as a metric, the relative discrimination capability of each feature can be determined.

Classifier Training

The classifier most closely resembles the Kohonen network with one very significant difference. This implementation has a variable number of nodes which are created and deleted whenever the algorithm deems appropriate. The training system is nothing more than a serial computer executing a k-means classifier. The ideal run time system would be a three-dimensional

SIMD parallel processor implementation. Such an architecture could implement the nodes along the z-axis while x-axis and the y-axis would span the image region of interest. Currently, a two-dimensional SIMD architecture would be the most cost efficient implementation that could still maintain real time performance speeds. The three-dimensional SIMD approach would use the processors parallel dimensions to span the image region of interest, and it would then serially emulate the node functions within each processor cell.

The chosen training strategy begins with a classifier with no nodes defined. The first sample in the training set is chosen to be the initial node center for the first node. Subsequent samples are then tested to determine whether they are sufficiently close to this node. If the sample is close enough, it is included in the node. When a sample is absorbed into the node, it is used to update the node's mean and covariance statistics using the iteration formula:

$$\mu_i(n) = (x_i(n) + (n * \mu_i(n-1))/(n+1) \tag{1}$$

$$\sigma^2_{ij}(n) = ((x_i(n) - \mu_i(n-1)) * (x_j - \mu_i(n-1)) + n * \sigma^2_{ij}(n-1))/(n+1) \tag{2}$$

$$i = 1 \cdots NumberOfFeatures, j = 1 \cdots NumberOfFeatures, n = 0 \cdots NumberOfSamples$$

These statistics are computed for each possible class as well as for all classes within the node. The inter-class statistics are used to determine which node should be used to classify a given sample. The intra-class statistics are used to determine the appropriate class for a sample, given that this node will be the one making the determination.

After populating the node with all nearby samples, the sample list is then checked against all nodes to insure that every sample falls within the operating range of at least one node. If any sample is still not represented, another node is generated and that node is populated. This process is repeated until all samples are represented.

Once the nodes are produced, the inverse covariance matrices are computed. The classifier is then tested using the training samples to evaluate the self-organized model. Each node is then scored to determine its discrimination accuracy. The score is computed as the number of correct classifications divided by the total number of classifications.

After computing each node's score, the higher scoring nodes are then analyzed in an attempt to modify the feature weighting coefficients. In each high scoring node, the ratio of inter-class variance to the sum of the intra-class variances is computed. The average of this ratio is taken as the metric by which all feature weights will be scaled. These ratios are normalized to range from 0.0 to 1.0, since changing the weights influences the node operating ranges.

The final stage of the training process eliminates poor or unused nodes. If a node's score falls below a threshold percentage, it is eliminated and the region will be reclustered during subsequent iterations. This entire process is repeated until either a satisfactory score is reached or when a fixed number of iterations has expired.

Classifier Implementation

Once the training process is completed, the network is then implemented as a run time Kohonen–like system. All features that have near zero weighting coefficients can be eliminated from the entire design, since they have no influence. The remaining features are coded into the run time system. Figure 2 illustrates the neural network implementation of the system.

Input Feature Vector

Node 1   Node 2   ...   Node N – 1   Node N

Proximity
And
Class

Select Closest Node
and
Output its Classification

Output Classification
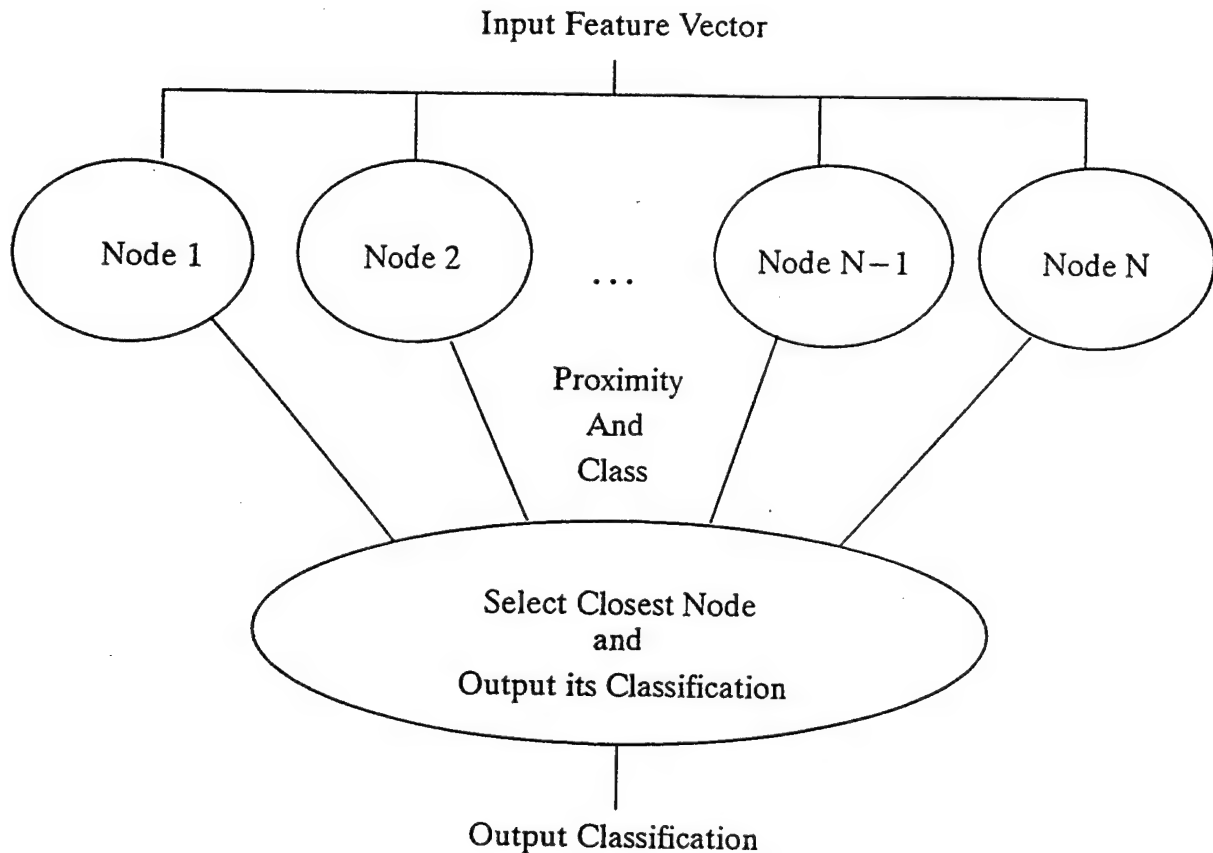
Figure 2 Kohonen Implementation of Image Segmenter

Training and Testing

A ground truthing program was designed using the Xview library routines to permit an operator to associate pixels with known classifications. This program reads an input image from disk, displays this image, and allows the operator to draw over any pixel he wished to label. The operator may then select any of eight colors to represent each class. Thus, the software is currently limited to eight or fewer class operation. After carefully labelling the desired pixels, the operator may then request that the program output the ground–truthed image. This image contains only pixels representing locations that the operator designated to specific classes. All other pixels are zero valued.

A second program then computes the feature vectors for the original imagery. This program reads the ground–truth image as well, and it outputs an ASCII text file containing the Cartesian coordinate of the pixel, the feature values for each feature, and the designated class. This file is maintained in ASCII for operator convenience. An operator can easily split the file into training and testing sets using various UNIX commands or even text editors.

Finally, a third program compacts an ASCII feature file into a binary representation that the classifier will use for training and testing.

Testing the classifier requires that a scoring mechanism be devised. The chosen scoring method evaluated the classifier based on each node individually. A node is scored based on the number of correct classifications divided into the number of samples classified by that node. Thus a node that only classifies one sample will either rate as a perfect score or as a zero score. Nodes that are not used rate as zero score, and are removed in successive iterations. The overall score, however, is computed as the total number of correct classifications divided by the total number of input samples. This penalizes the classifier for failing to classify any sample.

3. RESULTS AND CONCLUSIONS

The cluster–based classifier shows promise in classifying pixels for image segmentation applications. Early testing began with a primitive clustering strategy. This weighted all normalized features equally, and all clusters were spawned from any training vector that did not fit within any existing node. All nodes were also built with a fixed radius that was manually selected. During this testing, up to 80 percent classification accuracy was achieved on when testing and training was done on the same data (train A / test A). Approximately 75 percent accuracy was achieved when training and testing data were independent (train A / test B). The testing was performed on complex aerial photo imagery with three defined classes.

Improved clustering strategies raised performance to 83 percent (train A / test B) for the aerial imagery. These implementations began using weighting coefficients controlled by the scatter ratio. The first attempts computed new weight ratios directly from the scatter ratios, but unstable behavior was noted and the process tended to oscillate without significantly improving the clustering. The best strategies follow the basic steps of cluster, evaluate scatter ratios for high scoring nodes, then enhance high ranking features and penalize low ranking features. By repeating these steps iteratively, subsequent clusterings tended to show between three and ten percent increases in classification accuracy with no more than 15 iterations.

These results suggest that an adaptive feature weighting scheme can improve the classification accuracy of image pixels.

Feature Detection In Synthetic Aperture Radar Images Using Fractal Error

E. David Jansing, Darrel L. Chenoweth
University of Louisville
Department of Electrical Engineering
Louisville, KY 40292
502-852-6289
edjans01@starbase.spd.louisville.edu
dlchen01@starbase.spd.louisville.edu

John Knecht
Naval Air Weapons Center (NAWC)
China Lake, CA 93555
619-939-8734
knecht@llab.chinalake.navy.mil

Abstract--This paper discusses a technique that will enhance man-made features in SAR images. The technique uses a metric called fractal error. Developed by Cooper, et al. [1] for aiding photointerpreters in detecting man-made features in aerial reconnaissance images, this metric is based upon the observed propensity of natural image features to fit a fractional Brownian motion (*fBm*) model. Natural scene features fit this model well, producing a small fractal error. Man-made features, on the other hand, usually do not fit the *fBm* model well and produce a relatively large fractal error. Therefore, the fractal error is useful as a discriminant function for detecting man-made features in SAR imagery. The fractal error metric is defined, an approach to segmentating man-made objects in SAR images is discussed, and the results are presented in this paper.

TABLE OF CONTENTS

1. INTRODUCTION

In modern tactical air warfare, to assist the the pilot and improve the probability of mission success, mission planning tools are essential. However, even with comprehensive preplanning, the tactical situation may undergo unpredictable changes during the execution of the mission. Instead of returning to the aircraft carrier and re-planning a new mission, it is desirable to be able to modify the mission en route. To do this it becomes necessary to data link mission-related information to aid the pilot in making mission changes. Various forms of surveillance and reconnaissance imagery coupled with other intelligence information and digital maps are among the information types to be transmitted and displayed to the flight crew. Included in this mission-related data may be processed imagery of the revised target area from an off-board Synthetic Aperture Radar (SAR). The image data sent to the flight crew can be used to enhance terrain features, line-of-sight perspectives, and highlight expected target area features on cockpit displays. The algorithms for accomplishing this form an essential part of an integrated mission planning system to aid the pilot in redefining the mission en route.

This paper outlines initial research towards a

system that will enhance man-made features in SAR images as an aid for pilots in locating targets. It uses a metric called fractal error. While this metric has been shown effective with aerial photography [1], it has yet to be tested with SAR imagery. This paper presents results which indicate the fractal error metric is robust and will segment man-made objects from natural terrain in SAR images.

2. Previous Work

Few attempts have been made to address the problem of recognition of man-made objects in SAR images.

He and Wang [2] present a set of texture measures derived from a so-called texture spectrum and apply these measures to SAR imagery. The basis of the features suggested by He and Wang is that a texture image can be considered as a set of texture units which characterize the local texture information for a given pixel and its neighborhood. From these local texture units, an image of the global texture aspects can be mapped. Using the local texture units, the frequency distribution of the texture units is calculated. From this frequency distribution comes the term texture spectrum. He and Wang use this texture spectrum to extract information to form quantitative feature vectors which incorporate characteristics of the texture, such as black-white symmetry, geometric symmetry, degree of direction, orientational features and central symmetry. They compared their results to previous research using the co-occurrence matrix approach [3], [4], [5], [6], and [7]. The results indicate the measures they chose provide a more useful feature vector for segmenting SAR images.

Other authors have suggested using statistical methods as a means to segment SAR data [8], [9], [10]. In an article by Rignot and Chellappa [11], a model for the conditional distribution of polarimetric complex data is combined with a Markov random field representation to segment the SAR image. Their method describes a model for the a posteriori distribution of the region labels. Optimization of these region labels is defined as maximizing the a posteriori distribution of the region labels (maximum a posteriori (MAP) estimate).

Solberg, Taxt and Jain [10] developed a general model for multisource classification of remotely sensed data, such as optical, infrared (IR) and SAR. Their model is also based on Markov random fields and was developed for the fusion of optical images, SAR images and Geographic Information System (GIS) ground cover data. Their model uses the spatial class dependencies, also called spatial context, between pixel neighbors in an image and temporal class dependencies between images of the same scene at different times. They have tested their model on Landsat TM images, European Remote Sensing (ERS-1) SAR images and GIS ground-cover maps on agricultural scenes with good results.

Geometric feature extraction techniques have also been used to extract information from images. Methods such as the Hough transform [12], [13] can be useful. Selvage, Chenoweth and Gold [14] describe a modification to the chord transform for detecting geometric structural content within an image. This transform has been shown to be both scale, translation- and rotation-invariant. The most noticeable problem with the chord transform, however, is its computational expense.

In recent years, there has been progress in using metrics derived from fractal geometry in the analysis of natural textures

[15]. Pentland [16] used fractal-based descriptions of image textures to effectively characterize natural visual imagery. There has also been progress in applying fractal characteristics to basic image analysis, such as texture segmentation [17], [18]. Using a fractional Brownian motion model (fBm model), Stewart, et al. [19] demonstrate the application of fractal random process models as features in the analysis and segmentation of SAR imagery. The fractal dimension of natural textures, such as grass and trees, was computed and used as texture features in a Bayesian classifier.

Stewart discussed the application of metrics described by fractal geometry to provide accurate measures of "roughness" and "irregularity" in scale-invariant natural forms. Others [20], [21] have used the local fractal dimension for segmentation and analysis of infrared (IR) imagery. Likewise, Cooper [22] implemented a localized version of the fractal error measurement and fractal dimension for segmenting aerial images.

3. THE FRACTAL ERROR METRIC

It is well known that many textures and scenes can be modelled as *fractals*. A fractal, according to La Brecque [23], "has a rough shape to one degree or another made of parts which, when magnified, resemble the whole." Literature describing fractals often lacks precision when attempting to define what a fractal is. However, the reader of fractal geometry theory can turn to Falconer [24] for a detailed description of the properties of fractals.

The set F, according to Falconer, is a fractal if it has properties such as:

1. F has a fine structure, that is, detail on arbitrarily small scales.
2. F is too irregular to be described in tra-

ditional geometrical language, both locally and globally.
3. Often F has some form of self-similarity, perhaps approximate or statistical.
4. Usually, the "fractal dimension" of F (defined in some way, and there are several unique definitions) is greater than its topological dimension.
5. In most cases of interest, F is defined in a very simple way, perhaps recursively (e.g., the Julia or Mandelbrot Sets).

Fractals may occur in many different forms. Mandelbrot [25] was the first to define *fractional Brownian motion (fBm)*. Cooper [22] gives an excellent description of fBm; it will be summarized here.

fBm is defined in terms of the differences between successive samples of a function. Let $B_H(\mathbf{x})$ represent a fBm signal, where $\mathbf{x}$ is a vector containing E independent variables. Then the increment of the fBm signal is described as $\Delta B_H = B_H(\mathbf{x_2}) - B_H(\mathbf{x_1})$, where $\mathbf{x_1}$ and $\mathbf{x_2}$ correspond to two distinct measurements. The increment ΔB_H is normally distributed with a mean of zero and a variance proportional to the $2H$ power of the Euclidean distance between $\mathbf{x_1}$ and $\mathbf{x_2}$. The mean of the fBm increment is

$$E[B_H(\mathbf{x_1}) - B_H(\mathbf{x_2})] = 0. \qquad (1)$$

Likewise, the variance of the increment of the fBm is

$$Var[B_H(\mathbf{x_2}) - B_H(\mathbf{x_1})] = \sigma^2|\mathbf{x_2} - \mathbf{x_1}|^{2H}, \quad (2)$$

where σ^2 is the proportionality constant of the variance and H is the Hurst parameter, and must be strictly $0 < H < 1$. Note that when $H = 0.5$, then the fBm model reduces to the original Brownian motion model. The value of H can be used to describe the fractal dimension D as

$$D = E + 1 - H, \qquad (3)$$

where E is the Euclidean dimension (or, the number of independent variables). Small values of H produce high fractal dimensions and large values of H produce low fractal

dimensions. Combining the equations for mean and variance, while also taking into account the fractal dimension, we can obtain the following relationship

$$E[\|B_H(\mathbf{x_2}) - B_H(\mathbf{x_1})\|] = k|\mathbf{x_2} - \mathbf{x_1}|^H. \quad (4)$$

Equation (4) is the essence of the fractal error metric.

The regions observed in a SAR image may be combinations of many different textures. Each texture can be represented by its fractal dimension. When attempting to determine the fractal dimension, it is also useful to know how closely a texure element may be considered a fractal. Thus, measuring the error in estimating the fractal dimension will produce a useful metric to determine the "fractalness" of a feature in the image. A small error will indicate that an image feature fits the fractal model well, and thus can be considered a fractal. Conversely, a large error will indicate that the image feature fits the fractal model poorly and the feature is man-made.

We can mathematically define the fractal error computation as follows.

$$E[\|G[\mathbf{x_2}] - G[\mathbf{x_1}]\|] = k|\mathbf{x_2} - \mathbf{x_1}|^H \quad (5)$$

$$E[\|\Delta G_{|\Delta \mathbf{x}|}\|] = k|\Delta \mathbf{x}|^H, \quad (6)$$

where G is the image function and $\Delta \mathbf{x}$ is the distance measure within the region. Estimates of H and k can be found by using a linear regression. These estimates of $\overline{H}$ and $\overline{k}$ can then be used to calculate the fractal error using the equation

$$Error_{|\Delta \mathbf{x}|} = E[\|\Delta G_{|\Delta \mathbf{x}|}\|] - \overline{k}|\Delta \mathbf{x}|^{\overline{H}}. \quad (7)$$

Using a "center-oriented" window with n separate increment measurements of the distance $\Delta \mathbf{x}$, an RMS error can be computed

$$RMSError = \sqrt{\frac{1}{n} \sum_{|\Delta \mathbf{x}|} (Error_{|\Delta \mathbf{x}_i|})^2}. \quad (8)$$

The RMS error is the degree to which a pixel can be considered to be part of a fractal-like texture element.

4. FRACTAL ERROR ALGORITHM

Using the method described in the previous section, Cooper [22] developed an algorithm for calculating the fractal error for each pixel in a scene. The algorithm is as follows:

1. Define a 5x5, 7x7, or 9x9 sliding window.
2. Calculate each $\Delta \mathbf{x}$ for each pixel in the neighborhood of the sliding window relative to the pixel of interest.
3. Using linear regression, find the slope and the y-intercept for each unique set of $\Delta \mathbf{x}$ in the window from the equation $ln(E[\|\Delta G_{|\Delta \mathbf{x}|}\|]) = ln(k) + Hln|\Delta \mathbf{x}|$.
4. Thus, $\overline{H} = slope$ and $\overline{k} = exp(y - intercept)$ from the above relationship.
5. Then, from $Error_{|\Delta \mathbf{x}|} = E[\|\Delta G_{|\Delta \mathbf{x}|}\|] - \overline{k}|\Delta \mathbf{x}|^{\overline{H}}$, the fractal error is calculated for that unique set of $\Delta \mathbf{x}$.
6. The RMS error is then represented by $RMSError = \sqrt{\frac{1}{n} \sum_{|\Delta \mathbf{x}|} (Error_{|\Delta \mathbf{x}|})^2}$.
7. The RMS error for that pixel is saved, the window is moved, and the process is repeated over the entire scene.

5. SEGMENTATION OF MAN-MADE OBJECTS USING THE FRACTAL ERROR METRIC

The fractal error metric is useful for segmenting non-fractal image features using a simple thresholding algorithm on the fractal error image. Adapting a method by Otsu [26], [27], a threshold is chosen to separate the natural fractal-like objects from the man-made objects using a maximum likelihood estimator. Otsu developed this segmentation method on the statistics of the distribution of the gray level values in an image. Here, we use the distribution of the fractal error within the image instead. Consider an image with pixels having discrete, normalized fractal errors from $[1, L]$. The distribution of the fractal errors can be viewed

as a histogram, $h(g)$, $g = 1, \cdots, L$. This histogram is a graph of the frequency of certain fractal error values within the image. An image can be segmented into two separate classes, C_1 and C_2, (in this case, natural objects and man-made objects) by estimating a threshold level, denoted by t. This means that those pixels in the image belonging to class C_1 have a fractal error in the range of $[1, \cdots, t]$ and those pixels in class C_2 have a fractal error in the range of $[t + 1, \cdots, L]$. Thus, the following statistics can be calculated:

$$\omega_1(t) = \sum_{g=1}^{t} p(g) \tag{9}$$

$$\omega_2(t) = \sum_{g=t+1}^{L} p(g) \tag{10}$$

$$\overline{g}_1(t) = \sum_{g=1}^{t} gp(g) \tag{11}$$

$$\overline{g}_2(t) = \sum_{g=t+1}^{L} gp(g) \tag{12}$$

$$\sigma_1^2(t) = \sum_{g=1}^{t} (g - \overline{g}_1(t))^2 p(g) \tag{13}$$

$$\sigma_2^2(t) = \sum_{g=t+1}^{L} (g - \overline{g}_2(t))^2 p(g), \tag{14}$$

where $p(g)$ is the cumulative number of pixels with gray level value g.

The mean and variance for both classes combined is given by

$$\overline{g}_T = \sum_{g=1}^{L} gp(g) = \sum_{j=1,2} \omega_j(t)\overline{g}_j(t) \tag{15}$$

$$\sigma_T^2 = \sum_{g=1}^{L} (g - \overline{g}_T)^2 p(g)$$
$$= \sigma_W^2(t) + \sigma_B^2(t), \tag{16}$$

where

$$\sigma_W^2(t) = \sum_{j=1,2} \omega_j(t)\sigma_j^2(t) \tag{17}$$

and

$$\sigma_B^2(t) = \sum_{j=1,2} \omega_j(t)(\overline{g}_j(t) - \overline{g}_T)^2. \tag{18}$$

Note that the statistics $\sigma_W^2(t)$ and $\sigma_B^2(t)$ are the within-class and the between-class variance.

Consider the classes C_1 and C_2 to be normally distributed with unique means, but a common variance. The probability density function for each class is represented by the expression

$$f(g|C_j) = \frac{1}{(2\pi\sigma^2(t))^{1/2}}$$
$$exp(-\frac{(g - \mu_j(t))^2}{2\sigma^2(t)}), \tag{19}$$

where $j = 1, 2$. Kurita, et al. [27] showed that the likelihood function then becomes

$$L(G|\Theta; B(t)) = \prod_{i=1}^{N} \prod_{j=1}^{2} [\frac{1}{(2\pi\sigma^2(t))^{1/2}}$$
$$exp(-\frac{(g - \mu_j(t))^2}{2\sigma^2(t)})]^{\theta_i(t)}$$
$$= (2\pi\sigma^2(t))^{-N/2}$$
$$exp(-\frac{1}{2\sigma^2(t)}$$
$$\sum_{i=1}^{N} \sum_{j=1}^{2} \theta_{ji}(g_i$$
$$-\mu_j(t))^2), \tag{20}$$

where g is the set of gray levels (or in our case, fractal error values), θ is a vector padded with zeros and a single one located in the position corresponding to the class of the pixel, and $B(t)$ is the set of parameters $\mu_1(t), \mu_2(t), \sigma(t)$. Taking the logarithm of the above expression, we arrive at the log-likelihood equation,

$$l(G|\Theta; B(t)) = -\frac{N}{2}log(2\pi)$$
$$-\frac{N}{2}log(\sigma^2(t))$$
$$-\frac{1}{2\sigma^2(t)}[\sum_{i=1}^{N} \sum_{j=1}^{N}$$
$$\theta_{ji}(g_i - \mu_j(t))^2]. \tag{21}$$

The optimal parameters are obtained from this expression as

$$\widehat{\mu}_j(t) = \overline{g}_j(t) \quad (j = 1, 2) \qquad (22)$$

$$\widehat{\sigma}^2(t) = \sigma_W^2(t). \qquad (23)$$

Therefore, the maximum log-likelihood function is expressed as

$$\widehat{l}(G|\Theta; \widehat{B}(t)) = -\frac{N}{2}log(2\pi)$$
$$-\frac{N}{2}log(\sigma_W^2(t))$$
$$-\frac{N}{2}. \qquad (24)$$

It is not difficult to see that, barring the constant terms, the maximization of this log-likelihood function with respect to the threshold value t is simply the minimization of the function $\sigma_W^2(t)$. Therefore, the optimal thresholding for an image due to its fractal error distribution is the value of t that minimizes σ_W^2.

ing method. The fractal error metric, originally developed by Cooper [22], provides information regarding the "fractalness" of a pixel in an image. This measure has been shown to provide enough information to discern between natural image features and man-made objects. A thresholding technique, presented by Otsu [26], was modified to systematically choose a threshold to segment the man-made objects from the background terrain using the fractal error metric. While this method seems to work well for aerial photography, it has problems with the SAR imagery, most notably with the specular noise inherent to SAR. Future research is focused on finding an algorithm to choose a threshold that will more effectively segment the SAR images without noise.

6. RESULTS

We tested the man-made feature segmentation algorithm using SAR imagery provided by Naval Air Weapons Center (NAWC), China Lake. Figure 1 shows a SAR image with some small cultural objects, buildings and roads, located in a desert and mountain environment. Below it is the fractal error image (normalized from 0 to 255). The lighter areas in the fractal error image correspond to pixels with higher fractal errors; the darker areas correspond to lower fractal errors. Thus, lighter areas in the resultant image tend to indicate man-made object. The third image shows the image segmented using the method outlined in the previous section. Darker areas in the image correspond to man-made features in the SAR image.

7. CONCLUSIONS

A method of segmentation was presented using the fractal error metric and a threshold-
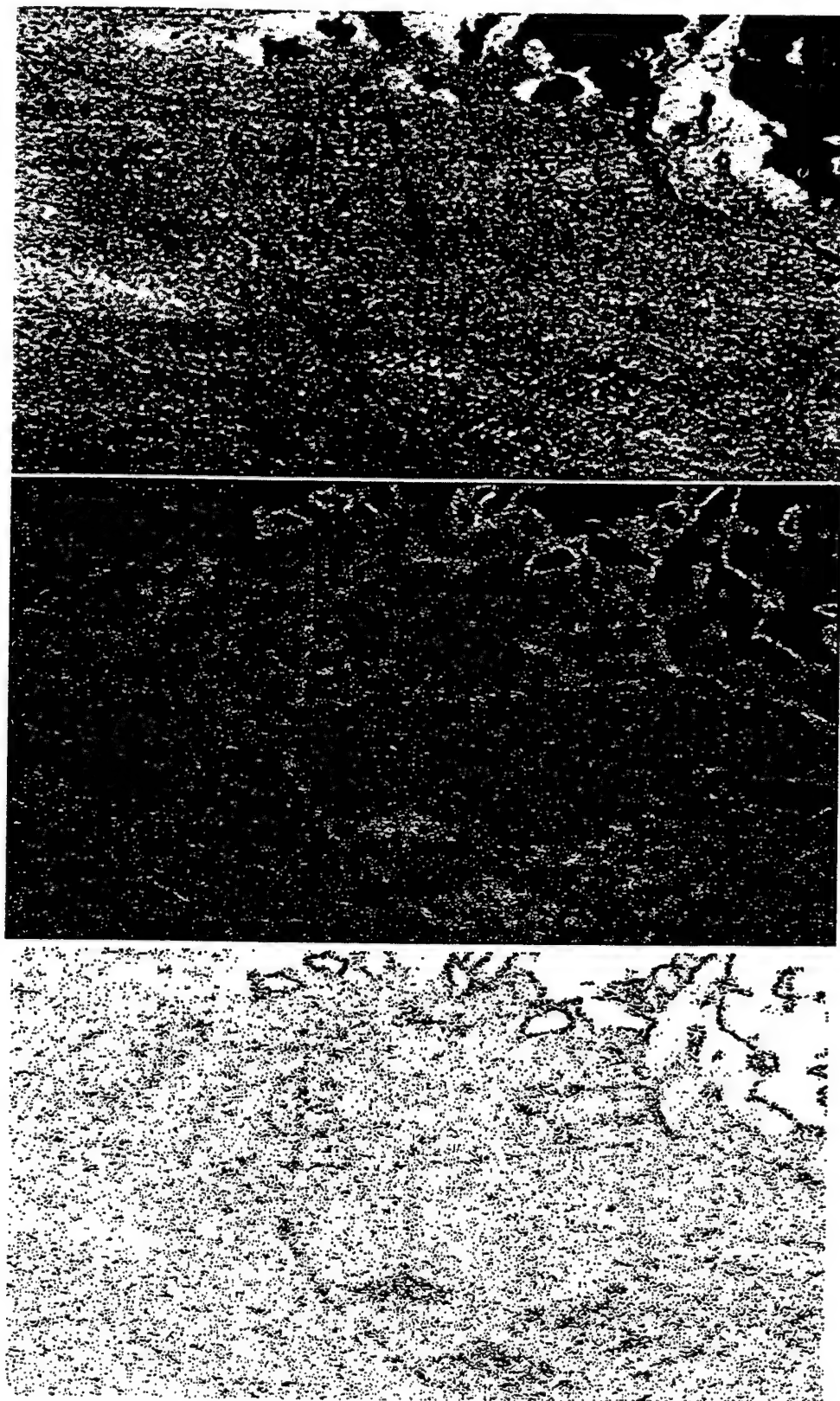
Fig. 1. SAR image, normalized fractal error image, and segmented image. Pixel size is approximately 1 square meter and the field of view is approximately 589 meters by 409 meters.
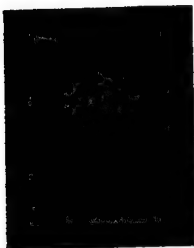
REFERENCES

[1] B. E. Cooper, D. L. Chenoweth, and J. E. Selvage, "Fractal Error For Detecting Man-Made Features In Aerial Images" *Electronics Letters*, Vol. 30, No. 7, pp. 554–555, 1994.

[2] D. H. He and L. Wang, "Texture Features Based On Texture Spectrum", *Pattern Recognition*, Vol. 24, No. 5, pp. 391–399, 1991.

[3] B. Julesz, "Visual Pattern Discrimination", *IRE Trans. Inform. Theory*, IT-8, pp. 84–92, 1962.

[4] A. Gagolowicz, "Visual Discrimination of Stochastic Texture Fields Based Upon Second Order Statistics", *Proc. 5th Int. Jt. Conf. Pattern Recognition*, Miami Beach, FL, pp. 786–789, 1980.

[5] R. M. Haralick, K. Shanmugan, and I. Dinstein, "Textural Features for Image Classification", *IEEE Trans. Syst. Man. Cybern.*, SMC-3, pp. 610–621, Nov. 1973.

[6] R. W. Conners, "Towards a Set of Statistical Features Which Measure Visually Perceivable Qualities of Texture", *Proc. Pattern Recognition Image Process. Conf.*, pp. 382-390, 1979.

[7] D. C. He, L. Wang and J. Guibert, "Texture Features Extraction", *Pattern Recognition Lett.*, Vol. 6, pp. 269–273, 1987.

[8] H. Derin, P. A. Kelly, G. Vezina, and S. G. Labitt, "Modeling and Segmentation of Speckled Images Using Complex Data", *IEEE Trans. Geosci. Remote Sensing*, Vol. 28, pp. 76–87, 1990.

[9] P. A. Kelly, H. Derin, and K. D. Hartt, "Adaptive Segmentation of Speckled Images Using a Hierarchical Random Field Model", *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. 36, pp. 1628–1641, Oct 1988.

[10] A. H. S. Solberg, T. Taxt, and A. K. Jain, "A Markov Random Field Model for Classification of Multisource Satellite Imagery", *IEEE Trans. Geosci. Remote Sensing*, Vol. 34, No. 1, pp. 100–113, Jan 1996.

[11] E. Rignot and R. Chellappa, "Segmentation of Polarimetric Synthetic Aperture Radar Data", *IEEE Trans. Image Processing*, Vol. 1, No. 3, pp. 281–300, Jul 1992.

[12] P. V. C. Hough, "Methods and Means For Recognizing Complex Patterns", US Patent 3 069 654, 1962.

[13] D. H. Ballard, "Generalized Hough Transform for Detecting Arbitrary Shapes", *Pattern Recognition*, Vol. 13, pp. 111–122, 1981.

[14] J. E. Selvage, D. L. Chenoweth, and V. Edward Gold, "Geometric Feature Extraction Using The Chord Transform", *IEEE Aerospace Conference*, Snowmass, CO, Feb 1996.

[15] B. B. Mandelbrot, *The Fractal Geometry of Nature*, San Francisco, CA, Freeman, 1983.

[16] A. P. Pentland, "Fractal-based Description of Natural Scenes", *IEEE Trans. Pattern Anal. Machine Intell.,*, PAMI-6, No. 6, pp. 661-674, 1984.

[17] J. M. Keller, S. Chen, and R. M. Crownover, "Texture Description and Segmentation Through Fractal Geometry", *Comput. Vision, Graph., Image Process.*, Vol. 45, pp. 150–166, 1989.

[18] S. Peleg, J. Naor, R. Hartley, and D. Avnir, "Multiple Resolution Texture Analysis and Classification", *IEEE Trans. Pattern Anal. Machine Intell.*, PAMI-6, No. 4, pp. 518–523, 1984.

[19] C. V. Stewart, B. Moghaddam, K. J. Hintz, and L. M. Novak, "Fractional Brownian Motion Models for Synthetic Aperture Radar Imagery Scene Segmentation", *Proceedings of the IEEE*, Vol. 81, No. 10, pp. 1511–1521, Oct 1993.

[20] B. Moghaddam, K. J. Hintz, and C. V. Stewart, "Fractal Image Compression and Texture Analysis", *19th AIPR Workshop*, SPIE, McLean, VA, Oct 1990.

[21] B. Moghaddam, K. J. Hintz, and C. V. Stewart, "Dimension and Lacunarity Measurements of IR Images Using Hilbert Scanning", *Int. Symp. on Optical Engineering and Photonics in Aerospace Sensing*, SPIE, Orlando, FL, Apr 1991.

[22] B. E. Cooper, *Fractional Brownian Motion For Representing Natural Image Texture*, Ph.D. Disseration, University of Louisville, 1994.

[23] M. La Brecque, "Fractal Applications", *Mosaic*, Vol. 17, No. 4, pp. 34–48, Winter 1986/87.

[24] K. Falconer, *Fractal Geometry-Mathematical Foundations and Applications*, New York: John Wiley and Sons, Ltd., 1990.

[25] B. B. Mandelbrot, *Fractals: Form, Chance and Dimension*, San Francisco: W. H. Freeman and Co., 1977.

[26] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms", *IEEE Trans. on Systems, Man, and Cybernetics*, SMC-9, No. 1, pp. 62–66, Jan 1979.

[27] T. Kurita, N. Otsu, and N. Abdelmalek, "Maximum Likelihood Thresholding Based On Population Mixture Models", *Pattern Recognition*, Vol. 25, No. 1, pp. 1231–1240, 1992.

E. David Jansing received his B.S. degree from the University of Louisville in 1991; and the M.Eng. degree in Electrical Engineering also from the University of Louisville in 1992. He is currently a Ph.D. candidate and Teaching Assistant at the institution. His research interests include Image Processing, Computer Vision, Global Optimization (including Genetic Algorithms), and Artificial Intelligence. He is a member of Eta Kappa Nu and IEEE.

Darrel L. Chenoweth is Professor and Chairman of Electical Engineering at the University of Louisville. He joined the University of Louisville in 1970 after completing his Ph.D. degree at Auburn University. He has been involved in image processing and pattern recognition research sponsored by the Naval Air Warfare Center and the Office of Naval Research since 1981. He is a Fellow in the Institution of Electrical Engineers.

John Knecht is an electrical engineer with Naval Air Warfare Center in China Lake, CA. He is working in mission planning systems.

4. REFERENCES

[1]   Kohonen, T., "Self-organized Formation of Topologically Correct Feature Maps,"
      Biological Cybernetics, 43, 59–69.

[2]   Rosenfeld, A. and Kak, A., *Digital Picture Processing*, Vol. 2, Academic Press, 1982.

[3]   Cooper, B. E., *Fractal Brownian Motion Representing* Natural Image Texture, Ph.D.
      Dissertation, University of Louisville, 1994.

EDGE ENHANCEMENT USING THE FRACTAL ERROR METRIC

E. David Jansing, Brian S. Allen, Darrel L. Chenoweth

University of Louisville
Department of Electrical Engineering
Louisville, KY 40292
502-852-6289
edjans01@starbase.spd.louisville.edu
bsalle01@starbase.spd.louisville.edu
dlchen01@starbase.spd.louisville.edu

ABSTRACT

This paper outlines a novel technique that will detect and enhance edge features in industrial images. While originally developed as an aid for pilots in locating their targets, this method is generally applicable to any type of image. The metric is based on the observed propensity of natural image features to fit a fractional Brownian motion (fBm) model [1], [2]. Because of their irregularity, edges do not typically fit the fBm model well. Thus, edges will have a higher fractal error than other features in the images, especially naturally-occurring texture. The fBm has been developed into a discriminant function to separate the edges from the other features in images. Results are presented and a comparison is made to the Canny edge operator.

1. INTRODUCTION

Edge detection has long been the subject of research in image processing and computer vision. Widely used in industrial vision, edge detection can be implemented for parts inspection or for robot vision. The model for edge detection is shown in Figure 1 and is a two stage process.
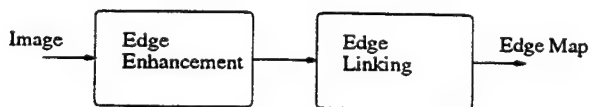
Fig. 1. The model for edge detection.

The first stage, edge enhancement, focuses on locating pixels in the image that can be considered a step edge. This process typically involves the initial detections of edge points using such techniques as the Laplacian of the Gaussian ([3], [4], [5]) or the gradient of the Gaussian [6]. False edges are eliminated from a set of criteria. For example, Canny [6] suggested thresholding edge points by comparing the magnitude of the gradient, an edge point candidate, against the magnitudes of its eight local neighbors.

The second stage, edge linking, attempts to link edges which might be broken. This break in the edges may be caused by shadowing or noise in the image. Some techniques for edge linking include the SEL algorithm [7] and the LINK algorithm [8]. These methods attempt to produce edges that are closed and connected over the image space.

This paper presents a novel method of edge enhancement, the first stage of the edge detection model. The method is based on the fractal error metric [1], [2]. It measures the error between the estimated fractional Brownian motion (fBm) model and the observed fBm. This model can be directly related to parts inspection, for example. A conveyer belt has a texture that may be classified as fractal in nature, with the part being non-fractal in nature. This metric effectively outlines the edges between this fractal texture and the part.

2. FRACTIONAL BROWNIAN MOTION

Fractional Brownian motion (fBm) is defined as the differences between successive samples of a function. First to define fBm, Mandelbrot [9] let $B_H(\mathbf{x})$ denote a fBm signal, where $\mathbf{x}$ is a vector containing E independent variables. Then the successive sample is described by

$$\Delta B_H = B_H(\mathbf{x_2}) - B_H(\mathbf{x_1}), \qquad (1)$$

where $\mathbf{x_1}$ and $\mathbf{x_2}$ correspond to two distinct measurements. The increment ΔB_H is Gaussian distributed with a mean of zero and a variance proportional to the $2H$ power of the Euclidean distance between $\mathbf{x_1}$ and $\mathbf{x_2}$. Thus,

$$E[B_H(\mathbf{x_1}) - B_H(\mathbf{x_2})] = 0$$
$$Var[B_H(\mathbf{x_1}) - B_H(\mathbf{x_2})] = \sigma^2 |\mathbf{x_1} - \mathbf{x_2}|^{2H}, \quad (2)$$

where σ^2 is the proportionality constant of the variance and H is called the Hurst parameter and must

be strictly $0 < H < 1$. Note that when $H = 0.5$, then the fBm model reduces to the original Brownian motion model. The value of H can also be used to describe the fractal dimension of the observation as

$$D = E + 1 - H \qquad (3)$$

where E is the Euclidean distance (or, the number of independent variables in the system). It is easy to see that small values of H produce high fractal dimensions and large values of H produce low fractal dimensions. An observation is said to be fractal if the fractal dimension D exceeds the topological dimension E [10]. Research has been done using the fractal dimension as a discriminant function in locating edge pixels (e.g., [11]).

Combining the equations for mean and variance produces the following relationship

$$E[|B_H(\mathbf{x_2}) - B_H(\mathbf{x_1})|] = k|\mathbf{x_2} - \mathbf{x_1}|^H, \qquad (4)$$

where k is the proportionality constant. Equation 4 is the essence of the fractal error metric. The regions observed in an image may be combinations of many different textures. Each texture can be represented by its fractal dimension. It has been shown [1], [2] that comparing the estimate of the fractal dimension to the actual data provides a useful metric to measure the "fractalness" of a feature in the image. A small error from the metric will indicate that an image feature fits the fractal model well, and thus can be considered fractal. On the other hand, a large error will indicate that the image feature fits the fBm model poorly and the feature is non-fractal in nature.

The error metric is defined from Equation 4 as

$$E[|G[\mathbf{x_2}] - G[\mathbf{x_1}]|] \;=\; k|\mathbf{x_2} - \mathbf{x_1}|^H \qquad (5)$$

$$E[|\Delta G_{|\Delta \mathbf{x}|}|] \;=\; k|\Delta \mathbf{x}|^H, \qquad (6)$$

where G is the image function and $\Delta \mathbf{x}$ is the distance measure within the region. Regions are typically defined as a center-oriented window of $5x5$, $7x7$ or $9x9$. Estimates of H and k can be found by using a linear regression scheme. These estimates of $\overline{H}$ and $\overline{k}$ can then be used to calculate the fractal error for the set $|\Delta \mathbf{x}|$ using the expression

$$Error_{|\Delta \mathbf{x}|} = E[|\Delta G_{|\Delta \mathbf{x}|}|] - \overline{k}|\Delta \mathbf{x}|^{\overline{H}}. \qquad (7)$$

Using the center-oriented window or region with n separate increment measurements of the distance Δx, an RMS error can be computed

$$RMS\ Error = \sqrt{\frac{1}{n} \sum_{|\Delta x|} (Error_{|\Delta \mathbf{x}|})^2}. \qquad (8)$$

The RMS error measures the degree to which a pixel can be considered to be a part of a fractal-like texture element. Edges will have higher fractal errors than non-edge features in the image. Thus, the fractal error metric makes an appropriate discriminant function for enhancing edges in real-world images.

3. EDGE ENHANCEMENT ALGORITHM

Using the method described in the previous section, a novel algorithm of enhancing edges has been developed:

1. Define a $5x5$, $7x7$ or a $9x9$ sliding window.
2. Calculate each $\Delta \mathbf{x}$ for each pixel in the neighborhood of the sliding window relative to the pixel of interest.
3. Using simple linear regression, find the slope and the y-intercept for each unique set of $\Delta \mathbf{x}$ in the window from the equation $ln(E[|\Delta G_{\Delta \mathbf{x}}|]) = ln(k) + Hln|\Delta \mathbf{x}|$.
4. Thus, $\overline{H} = slope$ and $\overline{k} = exp(y - intercept)$ from the above relationship.
5. Then, from Equation 7 the fractal error is calculated for that unique set of $|\Delta \mathbf{x}|$.
6. The RMS error is then calculated from Equation 8.
7. The RMS error for that pixel is saved, the window is moved, and the process is repeated over the entire scene.
8. A user-defined threshold τ selects the pixels that will represent edges and those pixels that will represent background. In other words, for $RMS[i][j]$ (where i and j are pixel coordinates in the image), a pixel is defined to be an edge pixel when $RMS[i][j] > \tau$ and a background pixel otherwise.

4. RESULTS

The algorithm was tested on an industrial image. This image shows a hypothetical part on a textured background (wood-grain). Figure 2 shows the test image, the result from enhancing the edges with the fractal error metric in a $5x5$ window, and the edges detected with the Canny edge operator with a threshold of 50. The test image was smoothed with the median filter before the edges were enhanced with the fractal edge enhancement algorithm. The image was smoothed with a $3x3$ Gaussian mask before edge enhancement and detection with the Canny edge operator. It is not difficult to see that the fractal error metric highlights the edges with the same accuracy as the Canny.

The Pratt figure of merit[12] provides a quantitative measure for the performance of edge detection algorithms. The figure of merit F penalizes an algorithm for missing valid edges, errors in localizing edges, and classification of noise as edges. The Pratt figure of merit is defined as follows:
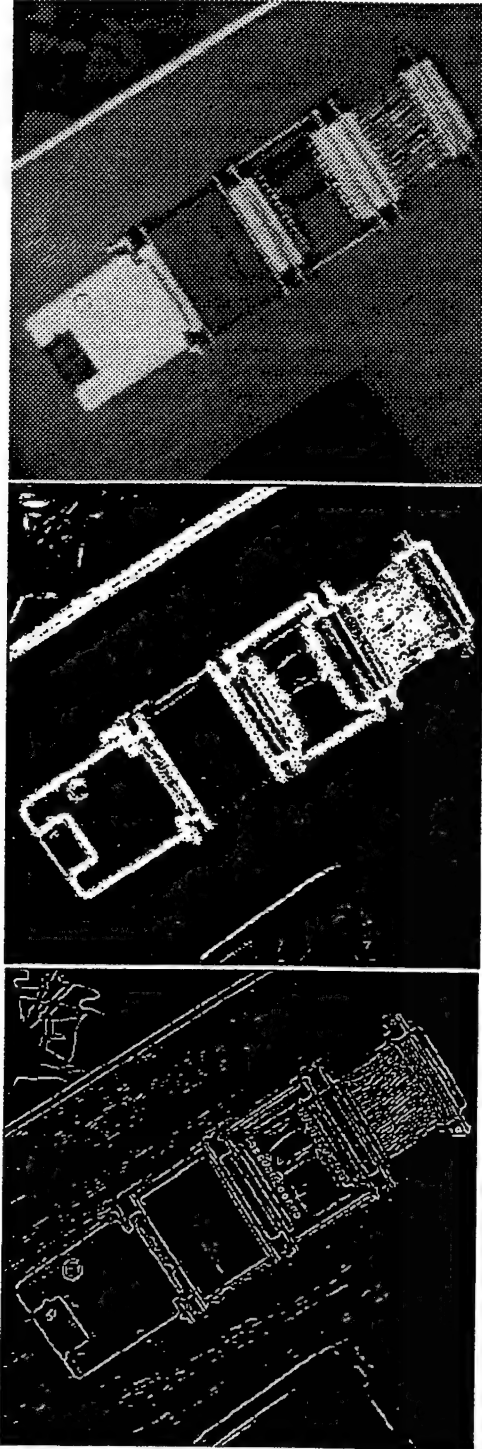
Fig. 2. The test image, the edges enhanced with the fractal error metric, and the edges detected with the Canny operator, respectively.

$$F = \frac{1}{I_L} \sum_{i=1}^{I_A} \frac{1}{1 + \alpha d^2}, \qquad (9)$$

where $I_L = max(I_I, I_A)$, I_I and I_A represent the num-

ber of ideal and actual edge points, respectively. α is a scaling constant used to penalize displaced edges, and d is the separation distance from an actual edge point to the nearest ideal edge point. The equation gives a merit of $F = 1$ for perfectly matched edges.

For most edge detectors, the Pratt figure of merit will decrease as the signal to noise ratio (SNR) decreases. Optimally, the Pratt figure of merit will decrease with a slope close to zero as the SNR decreases. To measure the performance as the SNR decreases, the edges detected from the original image with no added noise are used as the ideal edge points. Gaussian noise is then added to the original image to provide test images with various SNR's. The edges detected from the noisy images are used as the actual edge points, and the figure of merit is calculated. The fractal error metric algorithm and the Canny edge operator were performed on the test image with Gaussian distributed noise with a mean of zero and a standard deviation of σ. The fractal error metric performed as well as the Canny operator for images with little noise and performed much better for images corrupted heavily with noise. Again, both images were smoothed prior to edge enhancement (the fractal error metric images with the median filter and the Canny edge operator images with a $3x3$ Gaussian kernel). The fractal error metric had a threshold set for all of the images at $\tau = 130$ and the Canny edge operator was set at $t = 50$.
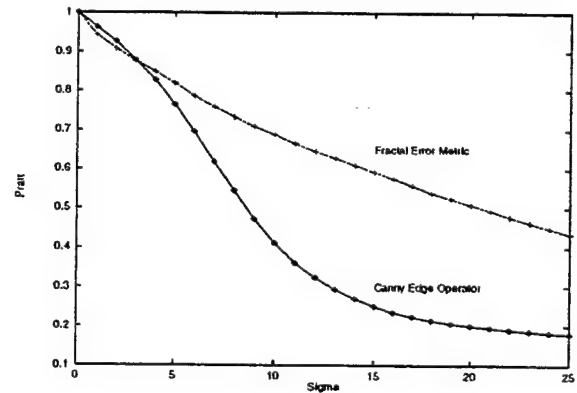
Fig. 3. Plot of the Pratt figure of merit for varying sigma.

5. CONCLUSIONS

A novel method for edge enhancement was presented. Using fractional Brownian motion (fBm) as a model, the estimates for the model are determined and the error is defined as the difference between the estimated

model and the actual model. This error metric provides a useful discriminant function in outlining edges in an industrial image. It was also shown that the fractal error metric performed better than the Canny edge operator for an industrial test image as the noise in the image increases.

REFERENCES

[1] B. E. Cooper, D. L. Chenoweth, and J. E. Selvage, "Fractal Error For Detecting Man-Made Features In Aerial Images", Electronics Letters, Vol. 30, No. 7, pp. 554–555, 1994.

[2] E. D. Jansing, D. L. Chenoweth, and J. Knecht, "Feature Detection In Synthetic Aperture Radar Images Using Fractal Error", Accepted to the 1997 IEEE Aerospace Conference, Snowmass, CO.

[3] D. Marr and E. Hildreth, "Theory of Edge Detection", Proceedings of the Royal Society, London, Vol. B-207, pp. 187–217, 1980.

[4] V. Berzins, "Accuracy of the Laplacian Edge Detector", Computer Vision, Vol. 27, pp. 195–210, 1984.

[5] J. J. Clark, "Authenticating Edges Produced By Zero-Crossing Algorithms", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-11, No. 1, pp. 43–57, Jan. 1989.

[6] J. Canny, "A Computational Approach To Edge Detection", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-8, No. 6, pp. 679–698, Nov. 1986.

[7] P. H. Eichel, E. J. Delp, K. Koral and A. J. Buda, "A Method For Fully Automatic Detection of Coronary Arterial Edges For Cineangiograms", IEEE Trans. Medical Imaging, Vol. MI-7, No. 4, pp. 313–320, Dec. 1988.

[8] A. A. Farag and E. J. Delp, "Edge Linking By Sequential Search", Pattern Recognition, Vol. 24, No. 5, pp. 611–633, 1995.

[9] B. B. Mandelbrot, Fractal: Form, Chance and Dimension, San Francisco: W. H. Freeman and Co., 1977.

[10] K. Falconer, Fractal Geometry, New York: John Wiley and Sons, Ltd., 1990.

[11] C. Chen, J. S. Daponte, and M. D. Fox, "Fractal Feature Analysis and Classification in Medical Imaging", IEEE Trans. Medical Imaging, Vol. 8, No. 2, June 1989.

[12] R. Jain, R. Kastuir, and B. G. Schnuck, Machine Vision, McGraw-Hill, Inc., 1995.

Neural and Genetic Approximations Of Fractal Error

Brian S. Allen, E. David Jansing, John E. Selvage, and Darrel L. Chenoweth
Department of Electrical Engineering
University of Louisville
Louisville, KY 40292
502-852-6289
bsalle01@starbase.spd.louisville.edu
edjans01@starbase.spd.louisville.edu
jeselv01@starbase.spd.louisville.edu
dlchen01@starbase.spd.louisville.edu

Abstract-- Fractal error is an image processing metric that can be used to locate man-made features in aerial images. The metric can aid photointerpreters in locating targets in aerial reconnaissance images. Fractal error was developed for this purpose by Cooper et al. [1]. Since the development, Jansing et al. [2] have shown that the fractal error metric also works well for extracting features in synthetic aperture radar (SAR) images. Jansing et al. [3] have also shown that the fractal error metric is useful for locating edge pixels in industrial images. The fractal error metric has a wide range of applications; however, some applications require real-time image analysis. The main disadvantage of the fractal error algorithm is that it can take several seconds to compute on large images. Therefore, it is desirable to create an approximation of fractal error to provide real-time image analysis. This paper presents two novel approximations of fractal error using a genetic algorithm and a neural network.

The results obtained using the approximations are compared with those obtained from the fractal error algorithm. Results from the neural network and the genetic algorithm are compared with one another. The neural network provides an accurate representation of fractal error, while the genetic algorithm does in fact preserve all of the desired features of the original fractal error image. The genetic algorithm has been shown to be computationally faster than the neural network.

Future work will consist of training a functional link neural network to improve computation time, and developing a genetic programming technique to evolve the mathematical structure and weights for a better fractal error approximation.

TABLE OF CONTENTS

1. INTRODUCTION

The mathematical nature of textures within images has generated much interest in the literature. Fractal dimension and lacunarity are often the focus of such interest. Notably, Pentland [4] used fractal dimension to analyze natural visual textures. However, little attention has been given to measuring the fitness of the fractal model itself. There has also been other work that has attempted to apply fractal characteristics and measures to basic image analysis, such as texture segmentation [5], [6]. Using a fractional Brownian motion (fBm) model, Stewart et al. [7] demonstrated the application of fractal random process models as features in the analysis and segmentation of SAR imagery. The fractal dimension of natural textures, such as grass and trees, was computed and used as texture features in a Bayesian classifier.

Stewart discussed how metrics described by fractal geometry provide accurate measures of "roughness" and "irregularity" in scale-invariant natural forms. Moghaddam et al. [8], [9] have used the local fractal dimension, a fractal metric, for segmentation and analysis of infrared (IR) imagery. Likewise, Cooper [10] implemented a localized version of the fractal error measurement as well as other fractal metrics for the segmentation of aerial imagery. Stein [11] introduced a slightly different fractal error metric, calculated by the covering method similar to the method suggested by Peleg [6]. Solka, Rogers and Priebe [12] introduced a power law signature similar to Stein's. There were significant differences, however. Stein designed a discrimination scheme employing

the slope and standard error of fit of the regression line, not unlike the algorithm Cooper implemented, which is outlined in this work. Solka, Rogers and Priebe used different features to estimate fractal error. Also, Stein's decision rules were heuristic in nature, whereas Solka, Rogers and Priebe proposed advanced density estimation techniques in an effort to fully characterize the decision surfaces which originate from their power law signatures.

In another paper, Rogers, Solka and Priebe [13] outlined a method to calculate fractal dimension using a parallel distributed processing (PDP) approach. *A priori* boundary information was incorporated into their covering method, improving their segmentation results.

Cooper et al. [1] developed a fractal error metric based on the observed propensity of natural image features to fit an fBm model. They used this feature in a statistical classifier to successfully segment regions in aerial reconnaissance images. Jansing et al. [2] also used this same feature to segment man-made objects in SAR imagery.

Upon examination, there are two important examples that provide motivation for fractal error analysis. The first example involves segmentation of cultural objects in SAR imagery, and the second is reliable edge detection in the presence of noise.

As an illustration of the need for segmentation, we will examine mission planning, a critical part of any tactical aircraft mission. To ensure the safety of the pilot and the success of the mission, extensive planning is required. Greater efficiency and consistency are achieved by supplementing pilot calculations with machine automation and intelligence. However, during the actual execution of the mission, the tactical situation may undergo unpredictable change. Instead of returning to the aircraft carrier and re-planning a new mission, it may be desirable to modify the mission en route. Therefore, it is necessary to data link mission-related information to aid the pilot with his new objective. Surveillance and reconnaissance imagery coupled with other intelligence information and digital maps can be presented to the flight crew. Included in this mission-related data may be processed imagery of the target area from an off-board sensor. These off-board sensors may provide IR imagery, SAR imagery, and optical imagery. SAR imagery is a likely sensor candidate. While highly specular and often difficult to interpret without enhancement, SAR is less susceptible to weather conditions and ambient illumination than IR. Thus, from all of the data sent to the flight crew, terrain features, line-of-sight perspectives, and expected target area features can be displayed in order to highlight the new mission aspects. The fractal error metric is described here in a method that provides useful detection of cultural objects. The resulting imagery could be used in on-the-fly mission adjustments.

As another example, edges typically do not fit the fractional Brownian motion (fBm) model well, due to their irregularity. Therefore, fractal error is also a practical edge enhancement operator. Jansing et al. [3] developed an algorithm utilizing fractal error to separate the edges from the other features in images.

This paper presents an approximation of Cooper's [10] fractal error measure using a genetic algorithm and a neural network. Computational expense is the unfortunate by-product of Cooper's algorithm. Since many applications require real-time computation to be practical, it is desirable to create an approximation of fractal error to provide real-time analysis, whether for the detection of cultural objects or edges.

2. FRACTAL ERROR

Definition Of A Fractal

It is well known that many textures and scenes can be modelled as *fractals*. A fractal, according to La Brecque [14], "has a rough shape to one degree or another made of parts which, when magnified, resemble the whole." It is also well known that literature describing fractals often lacks precision when attempting to define what a fractal is. However, the reader of fractal geometry and theory can turn to Falconer [15] for a detailed description of the properties of fractals.

Definition 1: The set F is a fractal, if it has the following properties:

1. F has a fine structure, that is, detail on arbitrarily small scales.

2. F is too irregular to be described in traditional geometrical language, both locally and globally.

3. F often has some form of self-similarity, perhaps approximate or statistical.

4. Usually, the "fractal dimension" of F (defined in some way, and there are several unique definitions) is greater than its topological dimension.

5. In most cases of interest, F is defined in a very simple way, perhaps recursively (e.g., the Julia or Mandelbrot Sets).

Fractal Dimension

How are fractals distinguished between one another? How does one measure the size of fractals? What measure can be used to compare and contrast fractals?
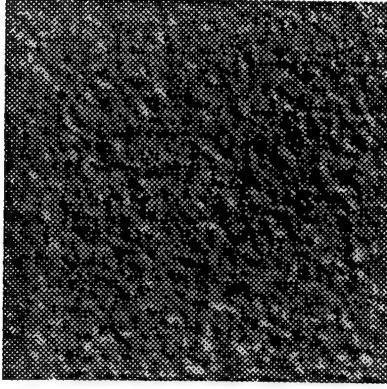
Fig. 1. Texture image of cork.

Fractal dimension is the measure that is generally used to distinguish between fractals, giving the fractals a measurement of "size". This numeric representation attempts to quantify a subjective quality which one might have about how densely the fractal occupies the space in which it exists.

Fractal dimension is just as difficult to define as fractals themselves. Mandelbrot [16], Falconer [15], Peitgen et al. [17] and Edgar [18] provide excellent discussions of many different fractal dimension definitions. Each fractal dimension definition has a distinct style. Although the definitions are all related, Peitgen [17] claims that some definitions make sense in certain cases, while other definitions may not be appropriate in the same case. Experience and heuristics prompt the selection of an appropriate fractal dimension definition, according to the application.

Lacunarity

Mandelbrot [16] defined another measure for fractals. *Lacunarity* describes the "holiness" ([19], pg. 236) of an occupied fractal lattice. The origin of the name lacunarity can be appreciated by looking at an image of cork in Figure 1. This image is part of a collection of texture images, presented by Brodatz [20]. From the Latin word "lacona," which means gap, lacunarity represents the gaps within a fractal structure. Thus, the percentage of spaces between the cork in Figure 1 is the measure of lacunarity. Practically, lakes or other natural objects within aerial images may be classified by using lacunarity as a feature measure.

Fractional Brownian Motion

Fractals may occur in many different forms. Mandelbrot [21] was the first to define *fractional Brownian motion* (fBm). Brownian motion refers to the erratic motion of small suspended particles, resulting from random collisions with other particles. Fractional Brownian motion is an extension of this model. Cooper [10] gives an excellent description of fBm; it will be summarized here.

The function of fBm is defined as the differences between successive samples. Let $B_H(t)$ represent a fBm signal, where t is a vector containing E independent variables. Then the increment of the fBm signal is described as $\Delta B_H = B_H(t_2) - B_H(t_1)$, where t_1 and t_2 are two distinct points in time. The measure ΔB_H is normally distributed with a mean of zero and a variance proportional to the $2H$ power of the Euclidean distance. The mean takes the form of

$$\mathrm{E}[B_H(t_2) - B_H(t_1)] = 0. \tag{1}$$

Likewise, the variance is defined as

$$\mathrm{Var}[B_H(t_2) - B_H(t_1)] = \sigma^2 |t_2 - t_1|^{2H}. \tag{2}$$

where σ^2 is the proportionality constant of the variance and H is the Hurst parameter, which must be strictly $0 < H < 1$. Note that when $H = 0.5$, the fractional Brownian motion model is equivalent to the classical Brownian motion model. The value of H can be used to describe the fractal dimension D as

$$D = E + 1 - H, \tag{3}$$

where E is the Euclidean dimension (or, the number of independent variables of $\mathbf{t}$). It is therefore easy to see that small values of H produce high fractal dimension and large values of H produce a low fractal dimension. Combining the equations for mean and variance, while also taking into account the fractal dimension, we can arrive at the following relationship:

$$\mathrm{E}[|B_H(t_2) - B_H(t_1)|] = k|t_2 - t_1|^H. \tag{4}$$

The above equation is the fundamental basis for the fractal error metric. The regions observed in an actual image may be a combination of many different textures. Each texture can be represented by its fractal dimension. However, before attempting to determine the fractal dimension, it is useful to know how well a region (or window) may fit the fractal model. Thus, measuring the error produced when estimating the fractal dimension will give a useful metric in order to determine the "fractalness" of a region in the image. A small error will indicate that a region fits the fractal model well, and thus can be considered fractal. Conversely, a large error will indicate that the region fits poorly into the fractal model and thus is probably not fractal and the fractal dimension measure is useless. Mathematically, this can be defined as

$$\mathrm{E}[|G[\mathbf{x_2}] - G[\mathbf{x_1}]|] = k|\mathbf{x_2} - \mathbf{x_1}|^H \tag{5}$$

$$\mathrm{E}[|\Delta G_{|\Delta \mathbf{x}|}|] = k|\Delta \mathbf{x}|^H \tag{6}$$

where G is the region or window in the image and $\mathbf{x}$ is the measured distances within the region. Estimates of H and k can be found by using a linear regression scheme,

$$\ln \mathrm{E}[|\Delta G_{|\Delta \mathbf{x}|}|] = \ln k + H \ln |\Delta \mathbf{x}|. \tag{7}$$

TABLE I
Fractal Error Algorithm

1.	Define a 5x5, 7x7, or 9x9 sliding window.
2.	Calculate Δx and $E[|\Delta G|]$ for each pixel in the neighborhood of the sliding window.
3.	Using linear regression, find the slope and the y-intercept for each unique Δx in the window from the equation $\ln(E[|\Delta G_{|\Delta x_i|}|]) = \ln(k) + H|\Delta x_i|$.
4.	Derive $\overline{H}$ = slope and $\overline{k}$ = exp(y-intercept) from the above relationship.
5.	Using $error_{|\Delta x_i|} = E[|\Delta G_{|\Delta x_i|}|] - \overline{k}|\Delta x_i|^{\overline{H}}$, calculate the fractal error for each unique Δx.
6.	Compute RMS error by $RMS \ Error = \sqrt{\frac{1}{n}\sum_{|\Delta x_i|}(error_{|\Delta x_i|})^2}$.
7.	Save RMS error for that pixel, move the window, and repeat the process over the entire scene.

These estimates, $\overline{H}$ and $\overline{k}$ can then be used to calculate the error with the following equation:

$$error_{|\Delta \mathbf{x}|} = E[|\Delta G_{|\Delta \mathbf{x}|}|] - \overline{k}|\Delta \mathbf{x}|^{\overline{H}}. \qquad (8)$$

Using a "center-oriented" window (i.e., a square window of NxN, where N is strictly odd), there will be five, nine, or fourteen error values, given the window is 5x5, 7x7, or 9x9 respectively. Thus, a cumulative error for the model can be given by the root mean square error

$$RMS \ Error = \sqrt{\frac{1}{n}\sum_{|\Delta \mathbf{x}|}(error_{|\Delta \mathbf{x}|})^2}. \qquad (9)$$

Thus, using the RMS error, it is easily determined whether or not a pixel with a surrounding 5x5, 7x7 or 9x9 region is fractal in nature.

3. Fractal Error Algorithm

Using the method described in the previous section, Cooper developed an algorithm to calculate the fractal error for each pixel in a scene. This algorithm is described in detail in Table I.

The following will outline the steps of the entire fractal error algorithm. Note that this example will produce a single number that will represent the error for the center pixel in relation to its neighbors. Figure 2 represents a sample 5x5 window. Since the localized neighborhood is 5x5, there are five unique distances in the window, as shown in Figure 3.

250	200	220	200	200
175	210	170	159	100
110	100	120	115	100
96	200	205	210	211
95	201	197	205	200

Fig. 2. A sample 5x5 window.

2.83	2.23	2.00	2.23	2.83
2.23	1.41	1.00	1.41	2.23
2.00	1.00		1.00	2.00
2.23	1.41	1.00	1.41	2.23
2.83	2.23	2.00	2.23	2.83

Fig. 3. The Euclidean distances over the 5x5 neighborhood.

Table II represents the absolute values of the differences of the gray scales over the unique sets of distances in the neighborhood. Thus, the gray scale value of each pixel that has a distance of 1.41 is subtracted from the gray scale value of the center pixel. Their absolute values are averaged to give $E[|\Delta G|]$ for each unique set of distances.

Using Equation 7, we can obtain estimates for the Hurst parameter and the proportionality constant, H and k respectively. These estimates can be found using linear regression [22]. If the linear model takes the form

$$y = \beta_0 + \beta_1 x, \qquad (10)$$

then estimates for β_1 (slope) and β_0 (y-intercept) are

TABLE II
Distances From The Center Pixel (Δx) And The Expected Value Of The Absolute Difference In Gray Scale Relevant To The Center Pixel ($E[|\Delta G|]$)

Δx	$\ln \Delta x$	$E[|\Delta G|]$	$\ln E[|\Delta G|]$
1.000	0.000	40.00	3.69
1.414	0.347	74.75	4.31
2.000	0.693	51.75	3.95
2.236	0.805	91.50	4.51
2.828	1.040	78.75	4.37

Δx	error
1.00	-6.06
1.41	18.4
2.00	-17.3
2.24	17.7
2.83	-5.9

defined as

$$\widehat{\beta_1} = \frac{\Sigma(x_i - \overline{x})y_i}{\Sigma(x_i - \overline{x})^2} \qquad (11)$$

$$\widehat{\beta_0} = \overline{y} - \widehat{\beta_1}\overline{x}, \qquad (12)$$

where $\overline{x}$ is the sample mean of x and $\overline{y}$ is the sample mean of y. In our particular case, x represents the "$\ln \Delta x$" term in Equation 7 and y represents the "$\ln E[|\Delta G|]$" term in the same equation. It easy to see that $\overline{x} = 0.577$ and $\overline{y} = 4.17$. The estimates of β_0 and β_1 can thus be calculated,

$$\widehat{\beta_1} = \frac{\sum_{i=0}^{5}(x_i - 0.577)y_i}{\sum_{i=0}^{5}(x_i - 0.577)^2} \qquad (13)$$

$$= 0.585 \qquad (14)$$

$$\widehat{\beta_0} = 4.17 - 0.577\widehat{\beta_1} = 3.83$$

The Hurst parameter estimate, $\overline{H}$, is equivalent to the slope of the linear model, that is, $\overline{H} = \widehat{\beta_1}$. The proportionality constant estimate, $\overline{k}$, is equivalent to the y-intercept of the linear model, $\overline{k} = \exp(\widehat{\beta_0})$. Therefore, $\overline{H} = 0.585$ and $\overline{k} = 46.1$.

Errors with respect to each unique distance set can be calculated with Equation 8. Table III shows the result of using Equation 8 in this example.

The overall RMS error, as defined by Equation 9, is 14.31. This number represents the "fractalness" of the center pixel relative to its neighbors. Decisions regarding the fractalness of the pixel are typically made in reference to the entire image. Thus, if the range of fractal errors is from 0 to 15, this pixel is not likely to be fractal in nature.

It is not difficult to see that the majority of computation comes from using the linear regression and calculating the errors from the estimated H and k. Two approximations have been developed to improve computation time without sacrificing accuracy. The first approximation uses a genetic approach, the second a neural approach.

4. GENETIC APPROXIMATION OF FRACTAL ERROR

In 1975, John Holland [23] introduced the publication *Adaptation in Natural and Artificial Systems*. In this publication, Holland integrated two themes that had persistently appeared in his studies: the portrayal of complex structures using simple representations (such as bit strings), and the ability to modify and improve such structures with simple transformations. Holland demonstrated that under the proper conditions, bit strings could "evolve" into improved bit strings. The improvement was dependent on the fitness of each member of the group of strings. Holland realized the tremendous potential of such an operation, and thus, initiated the study of *genetic algorithms*.

Holland discovered that the fundamental structures of genetics, chromosomes, are analogous to structures in other fields, such as types of goods in economic planning, strategies in gaming theory and functions in artificial intelligence. If the structures could be represented in a similar fashion, Holland posed the question of whether the same sort of operators that evolve chromosomes could be used to evolve optimal structures in the other fields. Holland demonstrated that, in fact, they do. Just as the operations of mutation and recombination adapt chromosomes to function in specific (and hopefully optimal) ways, the operations of production activities in economic planning or learning rules in artificial intelligence can adapt their respective structures.

In the same way chromosomes evolve over several generations, so do the structures in our genetic algorithm. Structures are allowed to sexually reproduce into the generation via some random operation according to the structure's fitness in the population. Typical fitness functions may include utility in economic planning, error functions in controls, payoff in gaming theory or comparative efficiency in artificial intelligence. Those individuals who are more fit than others (the lowest error in the case of controls or the highest payoff in the case of gaming theory) have a higher probability of reproducing into the next generation. As the total population evolves over generations, the best parts of fit structures combine with the best parts of other fit structures, creating structures that generate better fitness values. Holland [23] demonstrated that even in large, complicated search spaces, given certain conditions on the problem domain, the genetic algorithm would tend toward the global optima.

Much has been written about genetic algorithms and would be impossible to cite completely in this work. The reader should reference Holland [23], Goldberg [24], Davis [25], or Koza [26] for more detailed discussion of the theory behind genetic algorithms. While Holland, Goldberg and Davis primarily focused on genetic algorithms, Koza has spent much of his studies extending genetic algorithms to *genetic programs* (GP). Here the structures are actual computer code, typically in LISP or C. These structures of code are then allowed to evolve over several generations, until the needed

problem-specific code has been found. Using SPICE (Simulation Program with Integrated Circuit Emphasis), Bennett et al. [27] developed a GP to design a low-distortion, low-bias 60 dB (1000-to-1) amplifier with good frequency generalization. They contend that the performance of a GP can match or exceed human performance in some circuit design problems [28]. In 1996, Harris and Buxton [29] demonstrated that GP techinques can be used to evolve 1-D edge detectors.

To generate an approximation of fractal error, a mathematical structure is needed. Much of the information in calculating fractal error resides in the expected value of the absolute differences of the gray scales from the center pixel, grouped in uniform distances from the center pixel. This is expressed mathematically as $E[|\Delta G_{|\Delta x_i|}|]$. Weights are assigned to each of these expected values and all of the weighted expected values are added together to form the approximation of fractal error. This is expressed as

$$
\begin{aligned}
\widehat{FE} =\ & a_0 \ln(E[|\Delta G_{\Delta x = 2.82}|]) \qquad (15) \\
+\ & a_1 \ln(E[|\Delta G_{\Delta x = 2.23}|]) \\
+\ & a_2 \ln(E[|\Delta G_{\Delta x = 2.00}|]) \\
+\ & a_3 \ln(E[|\Delta G_{\Delta x = 1.41}|]) \\
+\ & a_4 \ln(E[|\Delta G_{\Delta x = 1.00}|]).
\end{aligned}
$$

The genetic approximation of fractal error (GAFE) evolves the weights a_0, a_1, ..., a_4 to try to estimate fractal error. The natural log of the expected values is used to maintain the linearity of the measure. As commonly used in the design of a GA, the probability for crossover is set at 0.7, the probability for mutation is set at 0.1, the probability for copy is set at 0.1, and the probability for migration is also set at 0.1. Migration, originally introduced by Potts et al. [30] in 1994, helps combat the problem of premature convergence. Potts et al. allowed for multiple sets of individuals to evolve using traditional genetic operators such as crossover and mutation. Each set of individuals acts as an independent population, like a tribe or village. The migration operator chooses individuals at random and moves them to another population. The interaction prevents inbreeding and promotes reproduction among those individuals that have good characteristics, that is, high fitness values.

The migration operator used in this work is an extension of the method proposed by Potts et al. [30]. Migration is implemented here by randomly generating a new individual and placing it in the new population. This randomness may allow for movement to a search space that the individuals in the previous population were unable to reach. This modification allows for the property of migration without the overhead of mulitple populations.

Each operator is chosen randomly from a uniform distribution. The population size and the maximum number

TABLE IV
SEVERAL TRIALS OF THE GAFE WITH VARYING POPULATION
SIZE AND GENERATIONS

Trial	Pop	Max Gen	SNR
A	100	10,000	1.27
B	500	100	1.52
C	500	2000	1.56
D	1000	2000	1.63
E	2000	250	1.74
F	10,000	90	3.82

of generations allowed for each simulation is variable. Fitness for each individual is determined from signal-to-noise ratio (SNR). SNR measures the ratio of original signal to noise introduced into a given test image. For our application, SNR is defined as

$$
SNR = \sum_{r=0}^{M}\sum_{c=0}^{N} I_m(r,c)^2 / \qquad (16)
$$
$$
\sum_{r=0}^{M}\sum_{c=0}^{N}(I_m(r,c) - I_o(r,c))^2
$$

where $I_m(r,c)$ is the measured image and $I_o(r,c)$ is the original signal for $M \times N$ sized images. The lower the value of SNR, the noisier the signal is. Thus, to evolve weights with a genetic algorithm, it is desirable to maximize the signal-to-noise ratio.

The GAFE is also written in parallel to ensure fast simulation times. The evaluation of fitness for each individual in the population consumes most of the run time for the GA itself. To parallelize the algorithm, each individual is evaluated for fitness separately, running on as many CPU's as are available. The parallel algorithm was executed on an SGI Onyx R10000 high-performance computer, having 16 CPU's.

Table IV shows several different trials of GAFE and their results. The varying population size and maximum number of generations allowed is also given. The training image for each of these simulations is shown in Figure 4.

Figure 7 illustrates the resulting image from each trial.

Each gene is comprised of 100 bits, 20 bits per weight. The first 19 bits are the magnitude, starting at 2^2 and ending at 2^{-16}. The 20th bit for each weight is a sign bit, allowing for positive and negative weights. Thus, the interval for each 20 bits is $(-8, 8)$. Table V shows the final weights for Trial F. Figure 8 shows the evolution of the population over several generations for Trial F.

The results are revealing. For very small populations, sized from 100 to 1000 genes, the GAFE is unable to isolate a search area to find a suitable set of weights; that

(a) Original Image

(b) Actual Fractal Error Image

Fig. 4. Training image for the genetic approximation to fractal error (GAFE).

TABLE V
FINAL WEIGHTS FROM GAFE, TRIAL F

$a_0 =$	0.0022583
$a_1 =$	0.096954
$a_2 =$	-0.034164
$a_3 =$	-0.0069427
$a_4 =$	-0.00093078

ion. Forcing a mathematical structure to find the approximation is restrictive. In the future, it may be prudent to investigate the evolution of a mathematical expression using a GP. This would expand the search space from merely searching the weights for $\widehat{FE}$ to searching the infinite space of reusable functions to find the approximation.

5. NEURAL NETWORK APPROXIMATION OF FRACTAL ERROR

Multilayer feedforward neural networks have been shown to be good function approximators, and are often referred to as universal approximators [31],[32]. Girosi and Poggio suggest that a large number of networks can approximate any continuous function arbitrary well [33]. Can fractal error be considered such a function? If so, can it be approximated with a neural network? Poggio and Girosi also state that most approximation schemes can be mapped into a neural network [34]. Since fractal error is an algorithm, it really cannot be considered a function. However, it can be modelled with a neural network, provided the network can learn from a given set of examples.

It is desirable to develop a neural network approximation of fractal error that will be consistent for all images. In other words, the network is trained using only one image. Once the final weights are obtained, they can be used to produce the desired output for any image. Since the fractal error algorithm calculates the fractal error of a pixel based upon neighboring pixels in a user defined window, a similar approach is used in designing the architecture of the neural network.

Consider the 5x5 window in Figure 2. With the fractal error algorithm, each pixel in the window is used to find the fractal error of the center pixel. Similarly, each pixel in the window will be used as an input to a multilayer feedforward network. Figure 5 shows a 5x5 window with a multilayer feedforward network. The pixels in the 5x5 window are labeled 1 through 25, which corresponds to the input neurons. The hidden layer consists of 25 neurons, and the output layer consists of only one neuron, whose output represents the fractal error of the center pixel. All neurons are unipolar continuous, and the ac-

is, it gets "stuck" on a local optimum. It is clear that larger populations fare better than smaller populations, regardless of the number of generations evolved. Even in the final experiment, where there are 10,000 genes in the pool, the final solution was found by chance. It is unclear that the optimal solution would be converged upon, even if the gene pool is allowed to evolve for many generations. This may be attributed to any number of factors, such as the actual mathematical structure of $\widehat{FE}$, the parameterization of the weights, or the choice of genetic operators and their probabilities. However, it is not difficult to see that the the weights evolved in Trial F produce a usable, if not noisy, image.

The mathematical structure of GAFE poses a unique problem. There is no generalized form of fractal error as it is defined above. A generalized form would make it possible to generate a type of convolution mask which could be used to compute fractal error in a timely fash-
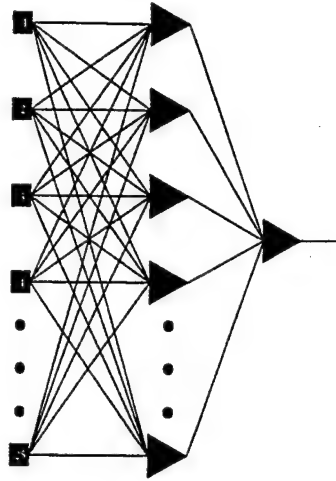
Fig. 5. Multilayer Feedforward Network for Approximating Fractal Error.

tivation function is defined by

$$f(net) = \frac{1}{1 + e^{-\lambda net}}. \qquad (17)$$

The network is trained using the error back-propagation algorithm. The training set, shown in Figure 6, consists of an input image and a desired output image. The input is an aerial image showing part of Alameda, CA. The desired output is obtained by computing the fractal error of the input image using the fractal error algorithm. The resulting image highlights the man-made features, including boats, buildings, roads, and runways.

The idea is to feed the input image into the neural network one 5x5 window at a time, and compute the output of the network. This output represents the approximated fractal error of the center pixel in the current 5x5 window. This value is compared to the the desired output, and the error is calculated. Weights are then adjusted according to the error back-propagation training algorithm (EBPTA). The process is repeated for each 5x5 window. The total accumulated error for the entire image determines when the training stops. If the total error is below a user defined maximum error, then the training stops. Otherwise, the error is reset, and the process starts over from the first 5x5 window.

Simply stated, input a 5x5 window into the neural network, compute the response of the output layer, and compare it to the desired output corresponding to the center pixel of the current 5x5 window. Repeat for each pixel until the total error for the entire images is sufficiently small. The weights are being "fine-tuned" so that, given any 5x5 window, the output of the neural network produces a number that nearly matches the value given by the fractal error algorithm. Table VI summarizes the training for the network.

Once the training is complete, the weights do not change.

TABLE VI
TRAINING THE NEURAL NETWORK TO APPROXIMATE FRACTAL ERROR

|	
1.	Read input image and normalize $[0.1, 0.9]$.
2.	Read desired output image and normalize $[0, 1]$.
3.	Initialize network parameters according to EBPTA.
4.	Input first 5x5 window and compute network output (network output represents the approximated fractal error of the center pixel in the current 5x5 window).
5.	Calculate error between desired output and network output according to EBPTA.
6.	Adjust the output and hidden layer weights according to EBPTA.
7.	If more inputs remain, input next 5x5 window, compute network output, and goto step 5. Else goto step 8.
8.	The training cycle is complete. If the error is below the user defined maximum error, terminate the training session. Output weights, the cycle number, and the error. If the error is above the maximum error, then reset the error and initiate a new training cycle by going to step 4.

Now, any image can be input into the network, and the fractal error for each pixel can be approximated using the steps outlined in Table VII.

The network was tested on several images. The results indicate that the neural network closely approximates fractal error. Although the network was trained using only the "Alameda" image, the final network is consistent for all of the test images. However, the network with 25 hidden layer neurons does not provide a fast approximation. In fact, computation time has slightly increased. When considering the number of mathematical operations required to compute the output of the neural network, it becomes clear why the computation time increases.

Each neuron has 25 inputs, which is multiplied by a weight and summed together. Each neuron also requires the evaluation of an activation function. There are 25 hidden layer neurons, and 1 output layer neurons, for a total of 26 neurons. The number of operations is $26 \times 25 = 650$ additions, $26 \times 25 = 650$ multiplications, and 26 activation function evaluations, for a total of 1326 operations required to approximate the fractal error of
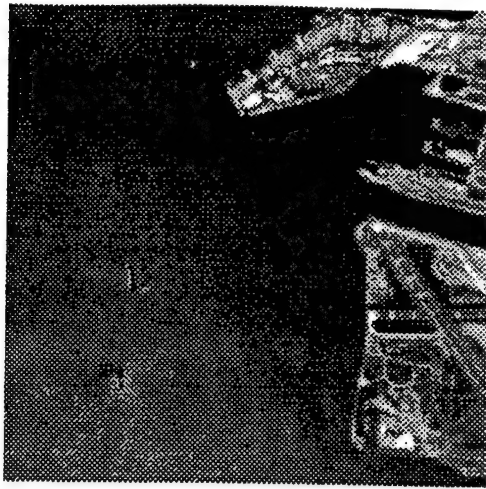
(a) Input Image
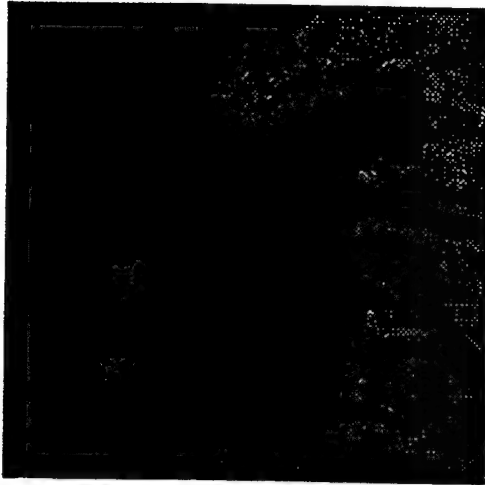
(b) Desired Output

Fig. 6. The Training Set for Learning to Approximate Fractal Error.

just one pixel. Obviously, the size of the network needs to be reduced.

Optimizing the Neural Network

Optimization, in the context of this paper, is two-fold. The neural network approximation needs to produce accurate results, and it also needs to provide them in a timely fashion. The network developed thus far does in fact produce the necessary accuracy. The RMS error is relatively small, as shown in Table VIII. However, the network cannot provide the results in a timely fashion.

Rather than try to improve accuracy that is already sufficient, efforts should be concentrated on trying to reduce

TABLE VII

FRACTAL ERROR APPROXIMATION VIA THE NEURAL NETWORK

1.	Read input image and normalize $[0.1, 0.9]$.
2.	Weights are final weights obtained from training.
3.	Input first 5x5 window. Compute and save network output (network output represents the approximated fractal error of the center pixel in the current 5x5 window).
4.	If more inputs remain, input next 5x5 window, compute and save network output, and repeat step 4. Else goto step 5.
5.	Normalize output $[0, 255]$ and write output image.

the size and complexity of the network in order to improve computation time.

If the computation time cannot be greatly reduced without losing accuracy, then this neural network architecture may not be suitable for the problem. Therefore, it seems reasonable to begin reducing the number of hidden layer neurons by at least half, or significant improvements in computation time will not be seen. Upon retraining the network with only 10 hidden layer neurons, the error function shows learning patterns very similar to the network with 25 neurons in the hidden layer. In fact, the new network very nearly matches the accuracy of the previous network, as shown in Figure 9.

Since nearly the same results were obtained with 10 hidden layer neurons, another reduction by half seems in order. Again, retraining the network with only 5 hidden layer neurons, the error function in Figure 9 shows learning patterns similar to the previous networks. This time, however, the error function does level off slightly sooner than before. The tradeoff between the loss in accuracy and the improvement in computation time must be considered. For the case of 5 hidden layer neurons, the loss of accuracy is minimal, while the improvement in computation time is significant.

6. RESULTS

Both approximation were tested on several images. The results for the "Washington D.C." and "Alameda" test images are shown in Figures 10 and 11. Table VIII shows the RMS error for both approximations relative to the fractal error algorithm. Table IX shows the computation times of the fractal error algorithm and both approximations for images of several different sizes.

7. CONCLUSIONS

Genetic and neural approximations have been developed in this paper. Both approximations provide usable results. While the neural network provides a more accurate representation of fractal error, it is clear that the genetic approximation preserves all the necessary features. Both approximations provide significant reductions in computation time, with the neural network being slightly faster. Currently, a genetic programming approach is being used to evolve a function to approximate fractal error. A functional link neural network is being implemented as well. It would be of great benefit to investigate combining the genetic and neural approaches discussed in this paper to improve the results.

(a) Trial A   (b) Trial B   (c) Trial C

(d) Trial D   (e) Trial E   (f) Trial F
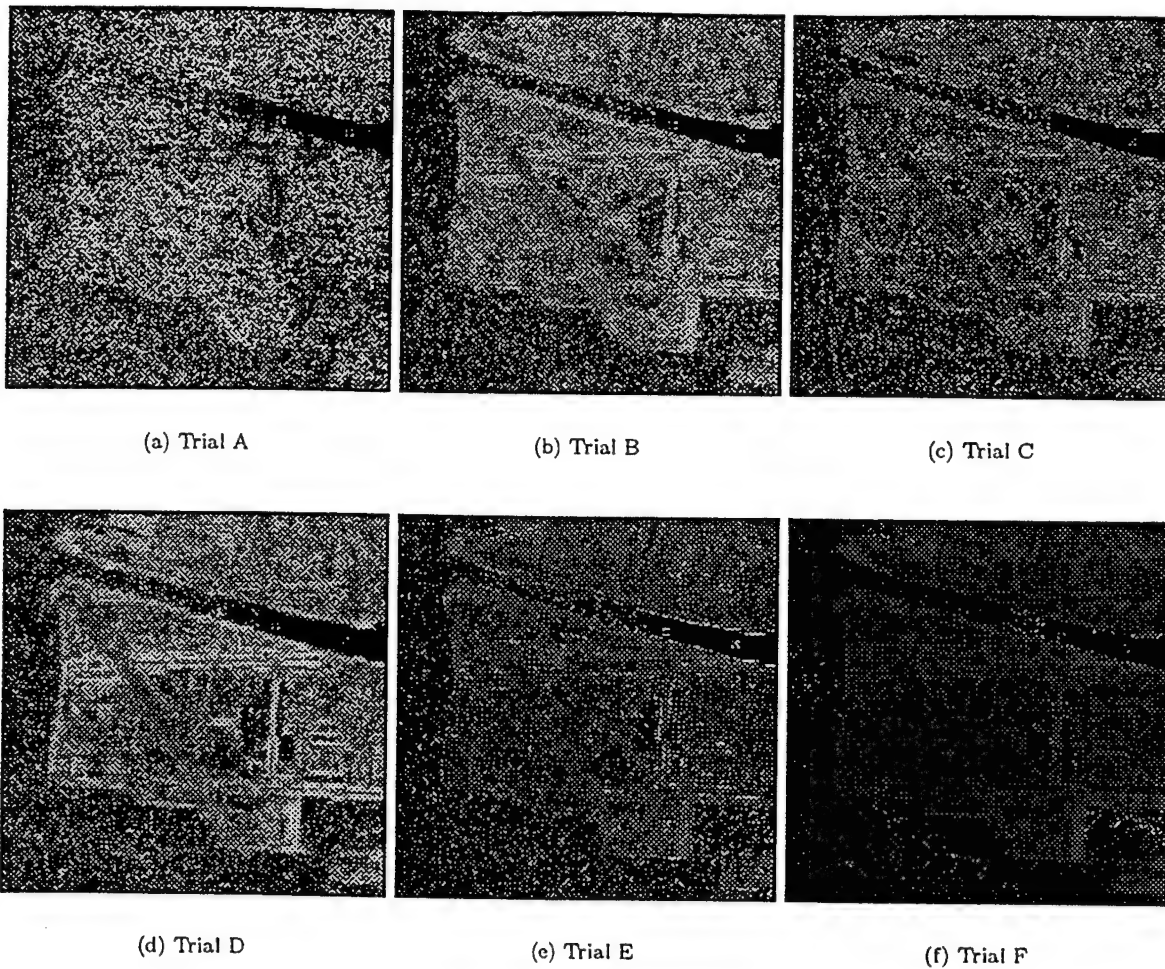
Fig. 7. Resulting $\widehat{FE}$ images from each GAFE trial described in Table IV.

TABLE VIII

RMS ERRORS OF TEST IMAGES WITH RESPECT TO ORIGINAL FE IMAGES

Image	Normalized RMS Error	
	GAFE	MFNN w/5 HLNs
Alameda	0.109	0.050
Washington D.C.	0.099	0.055

TABLE IX

COMPUTATION TIMES FOR ORIGINAL FE ALGORITHM, GAFE, AND MFNN

Size	Computation Time (seconds)		
	FE Algorithm	GAFE	MFNN w/5 HLNs
256x256	1.43	0.43	0.31
227x425	2.14	0.64	0.47
589x409	5.59	1.65	1.24
512x480	5.68	1.69	1.26
974x723	16.36	4.85	3.63
1267x941	27.74	8.22	6.13
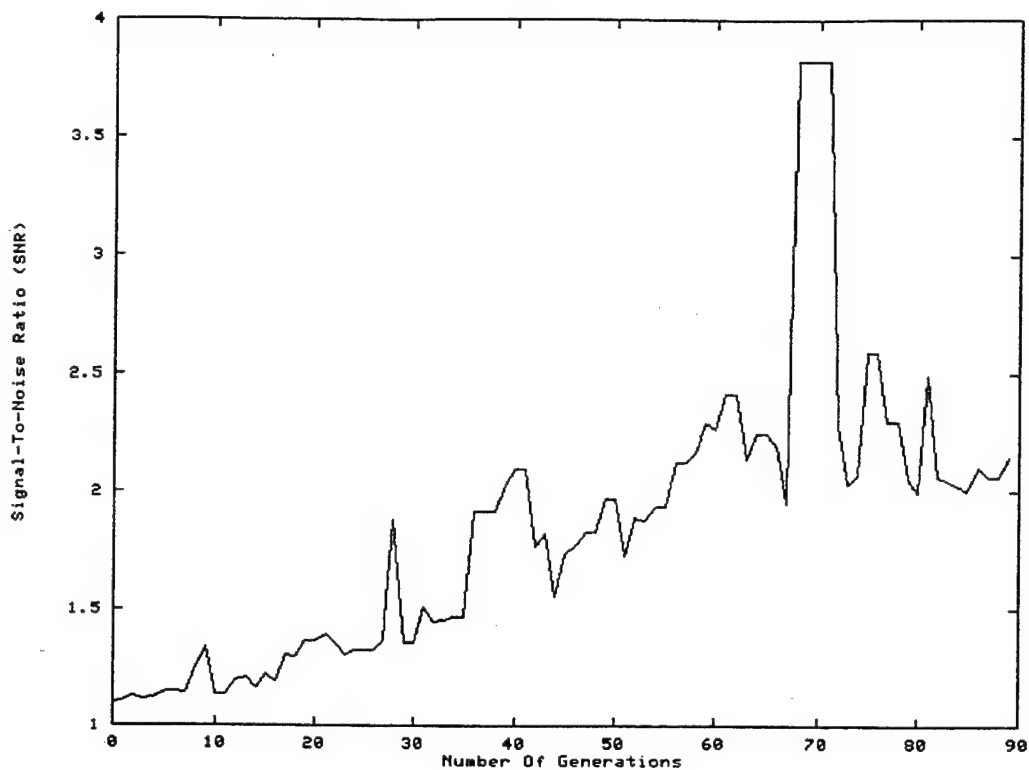
Fig. 8. Plot of Trial F with SNR vs. number of generations.
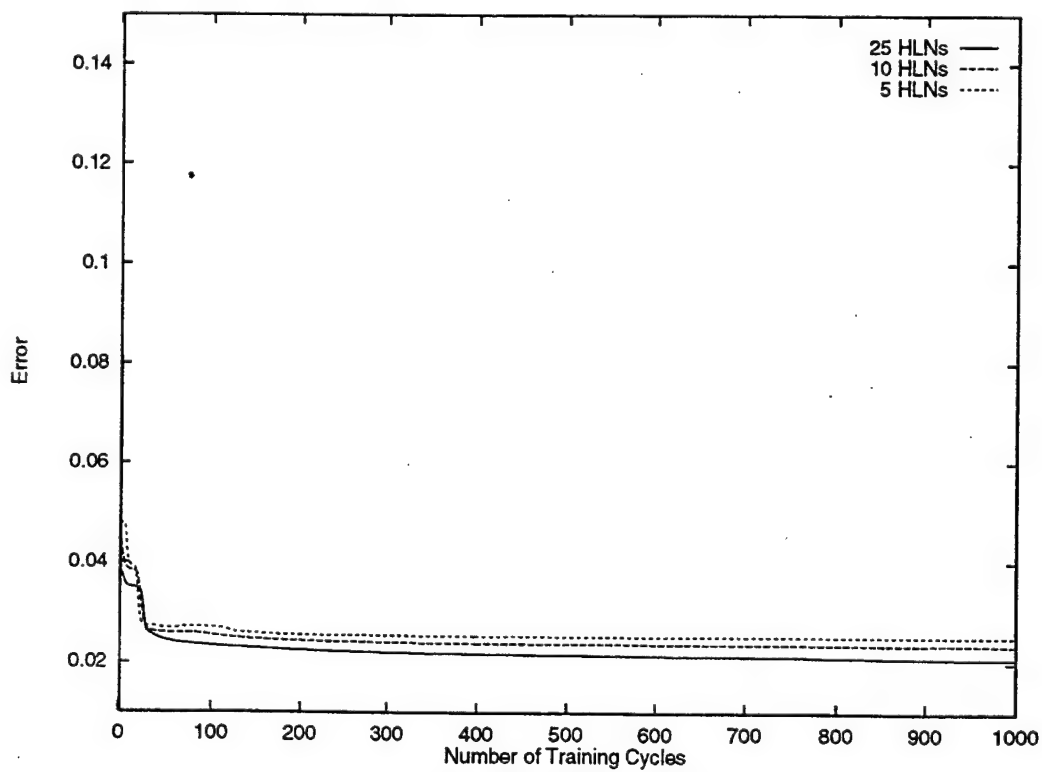
Fig. 9. Comparison of Error Functions for Different Number of Hidden Layer Neurons.
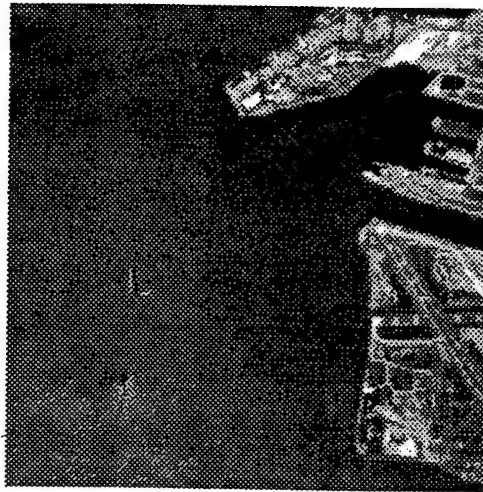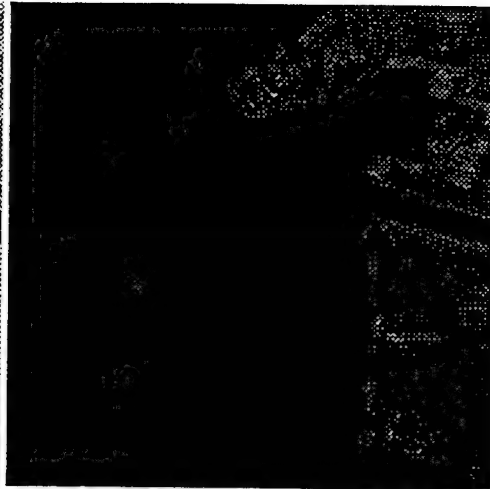
(a) Original
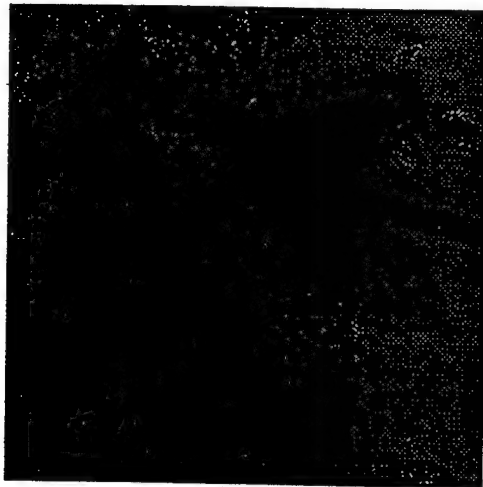
(b) FE Algorithm
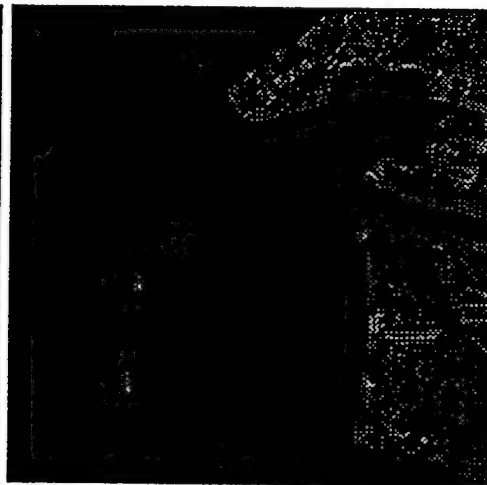
(c) GAFE

(d) MFNN w/5 HLNs

Fig. 10. (a) The original "Alameda" image (256x256), the outputs given by the (b) fractal error algorithm, (c) the genetic approximation, and (d) the multilayer feedforward network with 5 hidden layer neurons.
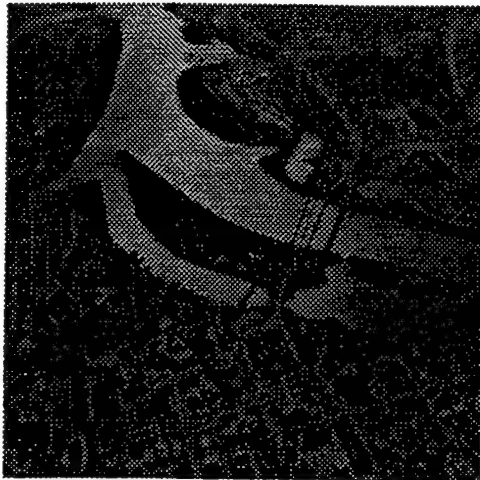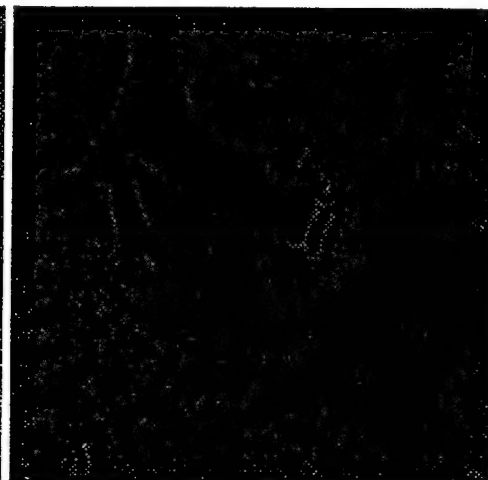
(a) Original

(b) FE Algorithm

(c) GAFE
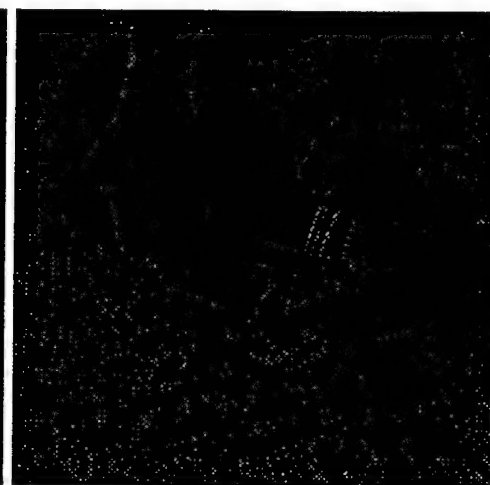
(d) MFNN w/5 HLNs

Fig. 11. (a) The original "Washington D.C." image (512x480), the outputs given by the (b) fractal error algorithm, (c) the genetic approximation, and (d) the multilayer feedforward network with 5 hidden layer neurons.

REFERENCES

[1] B. E. Cooper, D. L. Chenoweth, and J. E. Selvage, "Fractal error for detecting man-made features in aerial images", *Electronics Letters*, vol. 30, no. 7, pp. 554–555, 1994.

[2] E. D. Jansing, D. L. Chenoweth, and J. Knecht, "Feature detection in synthetic aperture radar images using fractal error", in *Proceedings of the IEEE Aerospace Conference*, 1997, vol. 1, pp. 187–195.

[3] E. D. Jansing, B. S. Allen, and D. L. Chenoweth, "Edge enhancement using the fractal error metric", in *Proceedings of the 1st International Conference on Engineering Design and Automation*, 1997.

[4] A. P. Pentland, "Fractal-based description of natural scenes", *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. PAMI-6, no. 6, pp. 661–674, 1984.

[5] J. M. Keller, S. Chen, and R. M. Crownover, "Texture description and segmentation through fractal geometry", *Computer Vision, Graphics and Image Processing*, vol. 45, pp. 150–166, 1989.

[6] S. Peleg, J. Naor, R. Hartley, and D. Avnir, "Multiple resolution texture analysis and classification", *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. PAMI-6, no. 4, pp. 518–523, 1984.

[7] C. V. Stewart, B. Moghaddam, K. J. Hintz, and L. M. Novak, "Fractional brownian motion models for synthetic aperture radar imagery scene segmentation", *Proceedings of the IEEE*, vol. 81, no. 10, pp. 1511–1521, October 1993.

[8] B. Moghaddam, K. J. Hintz, and C. V. Stewart, "Fractal image compression and texture analysis", in *SPIE 19th AIPR Workshop*, October 1990.

[9] B. Moghaddam, K. J. Hintz, and C. V. Stewart, "Dimension and lacunarity measurements of IR images usning hilbert scanning", in *SPIE International Symp. on Optical Engineering and Photonics in Aerospace Sensing*, April 1991.

[10] B. E. Cooper, *Fractional Brownian Motion For Representation Of Natural Image Texture*, PhD thesis, University of Louisville, Louisville, KY, 1994.

[11] M. C. Stein, "Fractal image models and object detection", in *Proceedings of the SPIE: Visual Communications and Image Processing II*, 1987, vol. 845.

[12] J. L. Solka, C. E. Priebe, and G. W. Rogers, "An initial assessment of discriminant surface complexity for power law features", *Simulation*, vol. 52, no. 5, 1992.

[13] G. W. Rogers, J. L. Solka, and C. E. Priebe, "A PDP approach to localized fractal dimension computation with segmentation boundaries", *Simulation*, vol. 65, no. 1, 1995.

[14] M. La Brecque, "Fractal applications", *Mosaic*, vol. 17, no. 4, pp. 34–48, Winter 1986/1987.

[15] K. Falconer, *Fractal Geometry–Mathematical Foundations and Applications*, John Wiley and Sons, Ltd., New York, NY, 1990.

[16] B. B. Mandelbrot, *The Fractal Geometry of Nature*, Freeman, San Francisco, CA, 1983.

[17] H.-O. Peitgen, H. Jurgens, and D. Saupe, *Chaos and Fractals: New Frontiers of Science*, Springer-Verlag, New York, 1992.

[18] G. A. Edgar, Ed., *Classics On Fractals*, Addison-Wesley, Reading, MA, 1993.

[19] B. H. Kaye, *A Random Walk Through Fractal Dimensions*, VCH, New York, 1989.

[20] P. Brodatz, *A Photographic Album For Artists and Designers*, Dover, New York, 1966.

[21] B. B. Mandelbrot, *Fractals: Form, Chance and Dimension*, W. H. Freeman and Co., San Francisco, 1977.

[22] L. J. Bain and M. Engelhardt, *Introduction to Probability and Mathematical Statistics*, PWS-Kent, Boston, 1992.

[23] J. H. Holland, *Adaptation In Natural And Artificial Systems*, The University of Michigan Press, Ann Arbor, 1975.

[24] D. E. Goldberg, *Genetic Algorithms In Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[25] L. Davis, *Genetic Algorithms and Simulated Annealing*, Morgan Kaufmann Publishers, Inc., London, 1987.

[26] J. R. Koza, *Genetic Programming: On The Programming Of Computers By Means Of Natural Selection*, The MIT Press, Cambridge, MA, 1992.

[27] F. H. Bennett, J. R. Koza, D. Andre, and M. A. Keane, "Evolution of a 60 decibel op amp using genetic programming", in *Proceedings of the 1st International Conference on Evolvable Systems: From Biology to Hardware*, October 1996.

[28] J. R. Koza, F. H. Bennett III, D. Andre, and M. A. Keane, "Four problems for which a computer program evolved by genetic programming is competitive with human performance", in *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, 1996.

[29] C. Harris and B. Buxton, "Evolving edge detectors with genetic programming", in *Proceedings of the 1996 Genetic Programming Conference*, 1996.

[30] J. C. Potts, T. D. Giddens, and S. B. Yadav, "The development and evaluation of an improved genetic algorithm based on migration and artificial selection", *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 1, January 1994.

[31] K. I. Funahashi, "On the approximate realization of continuous mapping by neural networks", *Neural Networks*, vol. 2, 1989.

[32] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators", *Neural Networks*, vol. 2, 1989.

[33] F. Girosi and T. Poggio, "Networks and the best approximation property", *Biological Cybernetics*, vol. 63, 1990.

[34] T. Poggio and F. Girosi, "Networks for approximation and learning", *Proceedings of the IEEE*, vol. 78, no. 9, 1990.

Brian S. Allen received his B.S.E.E. degree from the University of Louisville with a minor in Mathematics and plans to receive his M.Eng. degree in December of 1997. He is a member of IEEE and Eta Kappa Nu. His research interests include signal and image processing, neural networks, and artificial intelligence.

E. David Jansing received his B.S. degree from the University of Louisville in 1991, and the M.Eng. degree in Electrical Engineering also from the University of Louisville in 1992. He is currently a Ph.D. candidate and instructor at the institution. His research interests include image processing, computer vision, global optimization (including genetic algorithms), and artificial intelligence. He is a member of Eta Kappa Nu and IEEE.

John E. Selvage is a doctoral student at the University of Louisville, where he has received B.S. and M.Eng. degrees in Computer Science and Engineering. He is a former Senior Engineer at Martin Marietta Orlando Aerospace. He has done research in image processing, particularly object detection and classification algorithms, since 1986.
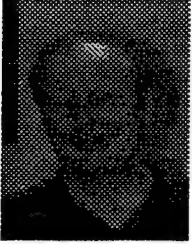
Darrel L. Chenoweth is Professor and Chairman of Electical Engineering at the University of Louisville. He joined the University of Louisville in 1970 after completing his Ph.D. degree at Auburn University. He has been involved in image processing and pattern recognition research sponsored by the Naval Air Warfare Center and the Office of Naval Research since 1981. He is a Fellow in the Institution of Electrical Engineers.

Fractal Error for Aerial Image Analysis

Brian E. Cooper [a],

Darrel L. Chenoweth, E. David Jansing and John E. Selvage [b]

[a] *Lexmark International, Lexington, KY 40550*

[b] *University of Louisville, Department of Electrical Engineering,
Louisville, KY 40292*

Summary

Although the fractal dimension, variants of the fractal dimension, and lacunarity have been used to analyze natural visual textures, the vast majority of research assumes that the observed image fits the fractal model. The error between the actual image and the fractal model should be considered. In this paper, a localized measure of the "fractal error" is developed for gray shade aerial image analysis. Two applications are examined: segmentation and cultural feature detection. Paired with the image intensity, the fractal error was found to be a suitable, general-purpose texture feature for segmentation. Also, the fractal error, without image intensity, demonstrated its ability to detect cultural image features.

Abstract

Although the fractal dimension, variants of the fractal dimension, and lacunarity have been used to analyze natural visual textures, the vast majority of research assumes that the observed image fits the fractal model. The error between the actual image and the fractal model should be considered. In this paper, a localized measure of the "fractal error" is developed for gray shade aerial image analysis. Two applications are examined: segmentation and cultural feature detection. Paired with the image intensity, the fractal error was found to be a suitable, general-purpose texture feature for segmentation. Also, the fractal error, without image intensity, demonstrated its ability to detect cultural image features.

Keywords: Aerial Image Analysis, Natural Texture, Fractals, Fractional Brownian Motion

1   Introduction

While there has been much work using fractal dimension, fractal dimension variants and lacunarity in image analysis, there has been little work in examining the validity of the model used in estimating fractal dimension. Most work considering the fractal dimension of an object in the image assumes that the image texture is, in fact, a fractal. It is useful to consider the possibility that an image texture may not accurately fit the fractal model. This paper will describe a metric based on the degree to which a pixel fits the fractal model. The first section of the paper outlines the background of fractals and fractal geometry. The motivation and theory of the fractal error metric is described in the second section. Finally, results are presented and some conclusions are drawn.

2   Background

2.1   Definition of a fractal

First popularized by Mandelbrot [1], fractals may occur in several forms, with applications ranging from data compression to fluid mechanics to par-

ticle aggregation and erosion. Appropriately, the meaning of the term *fractal* has widened from Mandelbrot's early tentative definition (in terms of the Hausdorff-Besicovitch dimension) to a broader set of properties. As described by Falconer [2], a fractal has an arbitrarily fine structure which may often be represented by some form of approximate or statistical self-similarity. Its fractal dimension, which may be defined in various ways, is usually greater than its topological dimension. Its irregularity, on both local and global scales, defies description by conventional Euclidean geometry. Such irregularity is common across a wide spectrum of natural processes, ranging from the microstructure of fractured materials to the formation of galactic clusters. In contrast to traditional geometry, fractal geometry readily accomodates the complexity and infinite detail present in nature.

2.2 Modeling natural terrains with fractals

Mandelbrot [3] noticed that natural terrain surfaces may be modeled by two-dimensional fractional Brownian motion with startling effectiveness. Introduced by Mandelbrot and Van Ness [4], fractional Brownian motion (sometimes abbreviated *fBm*) is a generalization of Brownian motion, which describes the complex, erratic movement of a particle, subjected to the collisions of the other paricles in its surrounding medium. Voss [5] established a link between the spectral distribution of fBm and the fractal dimension. Based on this, he used Fourier synthesis to generate natural terrains and even complete landscapes. Pentland [6] argued that natural terrain should follow a fractal model, since it is formed by fractal processes such as aggregation and erosion. Furthermore, he asserted that an image intensity surface of a fBm surface will also produce fBm, under normal lighting conditions. In other words, not only may terrain be modeled by fBm, but gray shade images of such terrain also.

2.3 Modeling image textures

Recognizing that fBm may be used to study images of natural terrain, Pentland [6] extended fBm to image textures in general. He found that fBm relief maps of varying fractal dimension correlated very closely with the human perceptual notion of roughness, a fundamental characteristic of texture. Estimating the fractal dimension from the Fourier transforms of local neighborhoods, Pentland distinguished the textures within a sampling of Brodatz textures and textures from natural scenes. The estimated fractal dimension was also used to segment the regions within images of a desert, San Francisco Bay and Mount Dawn. Keller et al. [7] developed a one-dimensional fBm model to distinguish between silhouettes of trees and mountains. With respect to

4

aerial image analysis, fractal dimension has been the dominant fractal characteristic. Most applications have studied the visual spectrum [8]–[12], although synthetic aperture radar and infrared imagery [13] have also been examined.

Image texture models are not restricted to using only the fractal dimension. In later work, Keller et al. [14] supplemented the fractal dimension with a lacunarity estimate to segment a set of Brodatz textures and a scene of trees. Likewise, Vehel [15] implemented an alternate form of each lacunarity and the fractal dimension to classify images of human lungs. Generalizing the normally scalar fractal dimension, Kaneko [16] used a 2×2 fractal dimension matrix to capture directional dependencies. The eigenvalue of this matrix was used to characterize Brodatz textures. Dennis and Dessipris [17] studied the rate of change of the fractal dimension as a texture signature across a range of pixel distances. In an effort to distinguish between natural and cultural (man-made) objects in aerial images, Cooper et al. [18] introduced the fractal error, which considers the degree to which an image region fits the fractal model.

3 The Model

3.1 Motivation

In spite of the variety of the fractal measures discussed above, nearly all of them rely upon one fundamental assumption: that the image texture is a fractal. Presumably, this situation results from Pentland's original assertion that images of natural terrain may be modeled by a fractal (specifically fBm) approach [6]. However, Pentland assumes that the scene's reflectance will be uniform. When this condition does not hold, the validity of the fractal model is questionable. In a study of the mathematical concerns of various algorithms to estimate the fractal dimension, Pruess [19] confirms this sentiment, stating that one of the essential attributes of an estimation algorithm is to "provide some statistical information as to the accuracy and sensitivity of the output." Before estimating the fractal characterisitics of an object, it is important to determine if such a model is appropriate. Furthermore, the degree to which an object fits the fractal model is an important characteristic itself.

3.2 Fractional Brownian motion

The predominant measure of a fractal is dimension. Therefore, the error in the estimation of the fractal dimension was examined, although the error in the estimation of other fractal characteristics could (and should) by considered.

5

Following the historical precedent set by Mandelbrot [3], Pentland [6], and others, the fractal dimension used here is based upon the H parameter of fractional Brownian motion.

Fractional Brownian motion (fBm) may be described by the properties of the signal's increments, which must be normally distributed with a mean of zero and a variance proportional to the $2H$ power of the Euclidean distance between the two measurements. Let $B_H(\mathbf{t})$ represent a fBm signal, where $\mathbf{t}$ is a vector containing E independent variables. Then the properties of the increments $\Delta B_H = B_H(\mathbf{t_2}) - B_H(\mathbf{t_1})$ are expressed below, where k represents the proportionality constant of the variance and H is a parameter between zero and one, exclusively $(0 < H < 1)$:

$$E[B_H(\mathbf{t_2}) - B_H(\mathbf{t_1})] = 0 \tag{1}$$

$$\mathrm{var}[B_H(\mathbf{t_2}) - B_H(\mathbf{t_1})] = \sigma^2 |\mathbf{t_2} - \mathbf{t_1}|^{2H}. \tag{2}$$

The fractal dimension D is related to the parameter H by the formula $D = N + 1 - H$, where N is the topological dimension (equivalently, the number of independent variables of the signal). Mandelbrot mentions this relationship as early as his 1975 work [3], but does not elaborate upon it. The interested reader should refer to Voss [5] for a lucid explanation.

Since $\mathrm{var}[Y] = E[Y^2] - E[Y]^2$, Equations (1) and (2) may be combined to give the expression

$$E[(B_H(\mathbf{t_2}) - B_H(\mathbf{t_1}))^2] = \sigma^2 |\mathbf{t_2} - \mathbf{t_1}|^{2H}. \tag{3}$$

However, the more common form is below, which may be derived through a transformation of random variables (see Appendix).

$$E[|B_H(\mathbf{t_2}) - B_H(\mathbf{t_1})|] = k|\mathbf{t_2} - \mathbf{t_1}|^{H} \tag{4}$$

When considering a two-dimensional signal, such as an image, there will be two independent variables $(N = 2)$. The notation $G(\mathbf{x})$ refers to such a signal, which will be compared to the fBm signal $B_H(\mathbf{t})$. Thus, $\mathbf{x} = (x_r, x_c)$, which specifies the discrete row and column coordinates. The function $G(\mathbf{x}) \in [0, \ldots, 255]$ is a discrete value representing the gray shade at location $\mathbf{x}$.

If $G(\mathbf{x})$ is assumed to be fBm, then the following equations hold for some H $(0 < H < 1)$ and k $(k > 0)$:

$$E[|G(\mathbf{x_2}) - G(\mathbf{x_1})|] = k|\mathbf{x_2} - \mathbf{x_1}|^{H} \tag{5}$$

$$E[|\Delta G_{|\Delta \mathbf{x}|}|] = k|\Delta \mathbf{x}|^{H}. \tag{6}$$

6

The distance between a pair of pixels is $|\Delta\mathbf{x}| = |\mathbf{x}_2 - \mathbf{x}_1|$, and the difference in gray shade between these pixels is $\Delta G_{|\Delta\mathbf{x}|} = G(\mathbf{x}_2) - G(\mathbf{x}_1)$. The average absolute gray shade difference, taken across pixel spacings of the same magnitude, denoted $|\Delta\mathbf{x}_i|$, will follow the exponential scaling described in Equation (6). A linear expression may be obtained by taking the logarithm of each side:

$$\ln(\mathrm{E}[|\Delta G_{|\Delta\mathbf{x}_i|}|]) = \ln(k) + H\ln(|\Delta\mathbf{x}_i|). \tag{7}$$

Using least squares linear regression, the estimated values $\overline{H}$ and $\overline{k}$ may be determined from a collection of pixel distances and their associated average absolute changes in gray shade.

3.3 Fractal error

However, $G(\mathbf{x})$ may or may not fit the fBm model. Although the above method will compute estimates for H and k, the validity of these values depends upon how well $G(\mathbf{x})$ corresponds to fBm. Thus, the error in the estimation should be considered. The error at a particular pixel spacing is

$$\mathrm{error}_{|\Delta\mathbf{x}_i|} = \mathrm{E}[|\Delta G_{|\Delta\mathbf{x}_i|}|] - \overline{k}|\Delta\mathbf{x}_i|^{\overline{H}}. \tag{8}$$

The root mean square (RMS) of these individual errors characterizes the overall error in the estimation, shown below. (Let n be the number of pixel distances considered.)

$$\mathrm{RMS\ error} = \sqrt{\frac{1}{n}\sum_i(\mathrm{error}_{|\Delta\mathbf{x}_i|})^2} \tag{9}$$

The RMS error measures the degree to which the observed signal fits the fBm model and is referred to here as the *fractal error*.

In a paper by Stein [10], a different fractal error measure is considered. Rather than using a set of gray shade differences in a localized neighborhood, Stein used a covering method similar to Peleg's method [11]. This method estimates the fractal dimension using bounded surfaces and morphology. The estimated dimension is then used to calculate the fit of the model to the actual data. Priebe et al. [12] also used the Peleg covering method to calculate a similar fractal error of each pixel in a scene. An F-test statistic was used to calculate the model fit, rather than the RMS error from the calculated error at each unique pixel spacing, as described above.

7

4 Example Implementation and Results

4.1 Developing a local fractal error measure

The implementation of the fractal error operator was oriented toward textures in gray shade aerial images, such as trees, grass, soil, water, shadows and buildings. In contrast to the high degree of intra-class homogeneity in Brodatz textures, the textures here demonstrate greater variation within their individual classes. The resolution of the images required a measurement within small, preferably overlapping neighborhoods of pixels. Window sizes from 5×5 through 17×17 were studied, with the smaller sizes giving the most favorable performance. Finally, the fractal error operator had to be quick and simple to use, so that it could be used interactively by a scene analyst who may not have a background in image processing.

In addition to capturing the textures better, the small window size also offered greater speed. Another factor was the choice of which pixel spacings to consider. For a window of size $w \times w$, there are $(w^4 - w^2)/2$ possible spacings. To reduce the number of computations, two approaches were examined. By considering only those pixel spacings in the horizontal and vertical directions, only $w^3 - w^2$ data points must be used. A further reduction results from a second technique. It examines all distances and angles within the window, but only with respect to the center pixel. There are $w^2 - 1$ such pixel spacings. Figure 1 illustrates the pixel spacings used within a 5×5 window for the center-oriented method. Intuitively, this approach seems to localize the measure more by emphasizing the center pixel, plus it strives to make the measure more directionally independent than the former approach.

4.2 Segmenting aerial imagery

Although gray shade is often the predominant feature for distinguishing regions in aerial images, gray shade alone may be inadequate. For example, the gray shade of the water in Figure 2(a) overlaps that of the foliage or trees. Thus, a combination of gray shade and fractal error was used with a Mahalanobis-distance statistical classifier. Stein [10] considered two different fractal error measures, as well as the fractal dimension, along with a set of heuristic decision rules in order to successfully analyze tactical aerial images. Each of the measures were dependent on user-selected thresholds that may introduce error in the segmentation. Priebe et al. [12] simplified the algorithm by considering only the estimated fractal dimension parameters.

Consider the shore scene (278×269) in Figure 2(a), containing water, foliage

and non-foliated land. The fractal error image in Figure 2(b), negated to improve visibility, shows that the water tends to have a low fractal error (light gray shade). The non-foliated land has a greater error, and the foliage has a medium to high error. Figure 2(c) shows the histogram of gray shade and fractal error, displayed as an image. The darkness of each point indicates the relative number of occurrences for that particular pair of gray shade and fractal error values. The distributions of the three classes foliage, water and non-foliated land are approximated by the ellipses superimposed on the histogram. Clearly, the combination of gray shade and fractal error can separate these classes. Using gray shade and fractal error to classify the pixels in the shore image, the resulting segmentation is shown in Figure 2(d), where the foliage is black, water is gray and soil is white. A neighborhood size of 5×5 was used in calculating the fractal error for these images.

This image was compared to a correctly segmented "ground-truth" image for these same three classes, shown in Figure 3(a). Compared against this ground-truth image, the gray shade and fractal error classification had an overall classification accuracy of 83.0%. Note that the majority of misclassifications are isolated pixels. Eroding all regions below the arbitrary size of ten pixels increases the classification accuracy to 89.0% and produces a significant improvement in the visual quality of the segmentation, as shown in Figure 3(b). Note that visual interpretation, as judged by a trained scene analyst, is typically the determinant of the quality of segmentation, since ground-truth imagery is seldom available.

In addition to the shore scene, other aerial images were considered. A scene of farm land (512×512) is shown in Figure 4(a). The light colored soil areas are clearly distinguishable solely on the basis of gray shade intensity. However, the grassy fields and foliage require a texture measure to be separated. Figure 4(b) shows the negated fractal error image, in which the foliage tends to have a larger error (darker gray shade) than the fields. In Figure 4(c), the image has been classified as foliage (dark gray), grass (light gray) or soil (white). Finally, Figure 5(a) introduces a mixture of natural content and cultural content (e.g., roads, buildings). (This image is also 512×512.) The lake in the upper left, the trees, the fields and cultural objects are generally separated in the negated fractal error image, shown in Figure 5(b). The classification in Figure 5(c) identifies four classes, from dark to light: water, foliage, open fields and cultural objects. Although ground-truth imagery was unavailable for these images, a visual comparison between the original and classified images indicates that gray shade and fractal error provide an effective way to segment the regions of interest in aerial imagery.

The fractal error should be large for regions which are not fractal. Since cultural or man-made objects often have a non-fractal appearance, the fractal error was examined as a way to identify cultural features. Over an appropriate range of sizes, cultural objects will appear to be non-fractal. Clearly, a uniformly colored man-made object which is much larger than the window size will appear to be fractal in the center with a low fractal dimension corresponding to its smooth, uniform gray shade. Its boundary will very likely have a non-fractal appearance, since it represents a transition between two different textures. Thus, the correspondence between the object size and the window size used by the fractal error operator will be significant.

This property was examined in the context of two images of a residential area (each 364 × 323) with a fairly high content of deciduous vegetation. The same scene was photographed in April and in October. This was done to evaluate the robustness of the metric relative to seasonal variations and changes in contrast and illumination that occurred at the different times. The primary man-made objects in the test images are about fifty to sixty pixels in size. The pair of images is shown in Figure 6. Figure 7 illustrates the fractal error of each of these images, using a 7 × 7 window size, thresholded so that features with large fractal errors appear black against a white background. It is obvious by comparing the original image and the binary thresholded fractal error images that the fractal error performed reasonably well in detecting man-made features. It was also fairly insensitive to changes in illumination, contrast, and season. Note the brighter lighting and greater contrast in the October image of Figure 6 compared to the April image, as well as the distinct increase of foliage in the October scene.

A selection of operators which were evaluated as candidates for detecting cultural features was tested on this pair of images. Cultural features usually possess clear edges, so three edge operators were included in the set. A texture operator previously studied for representing textures in aerial imagery, called "busyness," was also included. Originally developed by Rosenfeld and Kak [21], busyness measures the minimum average change in gray shade in each the horizontal and vertical directions within a local neighborhood of pixels. The fractal error with window sizes 5 × 5, 7 × 7 and 9 × 9 completed the set.

Quantifying an operator's accuracy for detecting the man-made content is difficult. The creation of a correctly segmented or "ground-truth" image involves a degree of subjectivity. Both the April and October gray shade images were used to create the ground-truth urban image shown in Figure 8 for the two classes "buildings" and "background." This image contains buildings which may have been obscured by foliage and shadows in one or possibly even both

10

of the original images. For example, the presence of a rooftop in a shadow may sometimes be implied by the geometry of the structure. Likewise, a building with very low contrast might be detected by a faint shadow. In some circumstances, objects were blurry even at higher resolutions, requiring a subjective determination whether the particular item was a portion of a building. Although the images were registered, there were slight misalignments of one or two pixels around the upper right and lower left corners of the image. The ground-truth segmentation attempted to compromise between the two images in such cases.

When comparing the fractal error, busyness and edge operators, each of the images processed by these operators was thresholded manually, while attempting to maximize the distinction between buildings and background. Although this task was performed carefully, it is inherently subjective. The fractal error and busyness images were compared directly to the ground-truth image. However, one would not expect the edge operators to match well to the filled objects of the correctly segmented image. At best, they can detect only the object boundaries, although this should be considered a valid detection of the object. Therefore, each edge operator was also applied to the ground-truth image, and the result was thresholded to a binary image. (While partially subjective, the determination of a threshold was much more obvious for this step than for the previous edge images.) Thus, the edge image of the ground-truth image was compared to the edge image of the original image for each of the three edge operators.

For each comparison, the percentage of correctly classified pixels was computed. The overall classification accuracy, based upon the relative number of pixels labeled as either buildings or background, was considered. However, this value can be misleading, since there were about five times more pixels labeled as background than as buildings. (For example, an image consisting entirely of background would still manage to produce an excellent overall classification accuracy above 80%.) Since the goal is the detection of buildings, the percentage of buildings classified as buildings might be a better means of comparison, although one could obtain perfect results with an image consisting entirely of buildings. However, the images were thresholded to produce meaningful results, not pathological situations, so this percentage may still be useful. An alternative criterion is the simple (non-weighted) average of the two matching classifications (i.e., buildings classified as buildings and background classified as background). Here, each the buildings and background classes are given equal importance.

The fractal error correctly classified 60% to 70% of the buildings, compared with 50% to 60% for the busyness and 10% to 35% for the edge operators. The average classification accuracy was roughly equal for the fractal error and busyness, between 65% and 70%. This measure dropped to between 50% to

60% for the edge operators. These results indicate that the visual appeal of the fractal error for identifying cultural features has quantitative support.

5 Conclusions

Few researchers have considered the error in their estimation of fractal characteristics. The error between the image data and the fractional Brownian motion model was therefore investigated as a texture descriptor, referred to here as the *fractal error*. The fractal error was developed as a localized texture feature, oriented toward the small, heterogeneous textures in gray shade aerial images.

Two applications were considered: region segmentation and cultural feature detection. The combination of gray shade and fractal error gave quick and accurate classification results. The fractal error was able to differentiate between regions whose overlapping gray shades prevented simple gray shade thresholding, such as water and foliage or foliage and fields. However, the fractal error will highlight sharp edges as high errors and has some degree of scale sensitivity.

Since cultural or man-made objects in an image will typically be non-fractal, the fractal error was applied to the detection of cultural features. The cultural content of the scene was located successfully using the fractal error. Compared both qualitatively and quantitatively with other simple operators, the fractal error provided superior performance in detecting cultural features.

The fractal error was found to be an effective feature for both of these applications of aerial image analysis. Because the observed data often deviates from the presumed model, the error should be considered in any expression of fractal characteristics. Further refinement could produce a more accurate measure of the "fractalness" of an image region.

6 Acknowledgements

Appendix

Theorem: Given a normally distributed random variable X with a mean of zero and variance σ^2, the mean of $|X|$ will be proportional to σ.

Proof: Let the probability distribution of X be labeled $f(x)$. From the given information,

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-x^2}{2\sigma^2}\right). \tag{10}$$

Also, let $Y = |X|$. Find the probability distribution of $g(y)$ of Y. (Note that $G(y)$ and $F(x)$ denote the cumulative distributions of $g(y)$ and $f(x)$, respectively.) $G(y)$ can be written as

$$G(y) = P(Y \leq y) = P(-y \leq X \leq y) = F(y) - F(-y). \tag{11}$$

Differentiating with respect to y produces

$$g(y) = f(y) + f(-y). \tag{12}$$

Since y must be non-negative, $g(y) = 0$ for $y < 0$. The distribution of $f(x)$ is symmetric about the origin, so $g(y) = 2f(y)$ for $y \geq 0$. Thus,

$$g(y) = \frac{2}{\sigma\sqrt{2\pi}} \exp\left(\frac{-y^2}{2\sigma^2}\right) \quad \text{for } y \geq 0. \tag{13}$$

By definition,

$$\begin{aligned}
E[Y] &= \int_{-\infty}^{\infty} y g(y) dy \tag{14}\\
&= \frac{2}{\sigma\sqrt{2\pi}} \int_0^\infty y \exp\left(\frac{-y^2}{2\sigma^2}\right) dy \\
&= \frac{-2\sigma^2}{\sigma\sqrt{2\pi}} \int_0^\infty \exp\left(\frac{-y^2}{2\sigma^2}\right) \frac{-y}{\sigma^2} dy \\
&= \left[\frac{-\sqrt{2}\sigma}{\sqrt{\pi}} \exp\left(\frac{-y^2}{2\sigma^2}\right)\right]_0^\infty \\
&= \sigma\sqrt{\frac{2}{\pi}}
\end{aligned}$$

Therefore, the mean of $|X|$ is proportional to the standard deviation of X, where the proportionality constant is $\sqrt{2/\pi}$.

13

References

[1] B. B. Mandelbrot, *Fractals: Form, Chance and Dimension* (W. H. Freeman and Co., San Francisco, 1977).

[2] K. Falconer, *Fractal Geometry–Mathematical Foundations and Applications* (John Wiley and Sons, Ltd., New York, 1990).

[3] B. B. Mandelbrot, Stochastic models for the Earth's relief, the Shape and the Fractal Dimension of the Coastlines, and the Number-Area Rule for Islands, *Proceedings of the National Academy of Sciences USA* **72** (1975) 3825–3828.

[4] B. B. Mandelbrot and J. W. Van Ness, Fractional Brownian Motions, Fractional Noises and Applications, *SIAM Review* **10** 4 (1968) 422–437.

[5] R. F. Voss, Fourier Synthesis of Gaussian Fractals: 1/f Noises, Landscapes, and Flakes, *State of the Art in Image Synthesis* Tutorial No. 10 SIGGRAPH (1983).

[6] A. P. Pentland, Fractal-Based Description of Natural Scenes, *IEEE Trans. Pattern Analysis and Machine Intelligence* **6** 6 (1984) 661–674.

[7] J. M. Keller, R. M. Crownover, and R. Y. Chen, Characteristics of Natural Scenes Related to the Fractal Dimension, *IEEE Trans. Pattern Analysis and Machine Intelligence* **9** 5 (1987) 621–627.

[8] G. G. Medioni and Y. Yasumoto, A Note on Using the Fractal Dimension for Segmentation, *IEEE Computer Vision Workshop* (April 30–May 2, 1984) 25–30.

[9] T. Peli, Multiscale Fractal Theory and Object Characterization, *Journal of the Optical Society of America* **7** 6 (1990) 1101–1112.

[10] M. C. Stein, Fractal Image Models and Object Detection, *Proceedings of the SPIE* **845** (1987) 293–300.

[11] S. Peleg, J. Naor, R. Hartely and D. Avnir, Multiple Resolution Texture Analysis and Classification, *IEEE Trans. Pattern Analysis and Machine Intelligence* **6** 4 (1984) 518-523.

[12] C. E. Priebe, J. L. Solka and G. W. Rogers, Discriminant Analysis in Aerial Images Using Fractal Based Features, *Proceedings of the SPIE* **1962** (1993) 196–208.

[13] C. V. Stewart, B. Moghaddam, K. J. Hintz, and L. M. Novak, Fractional Brownian Motion Models for Synthetic Aperture Radar Imagery, *Proceedings of the IEEE* **81** 10 (1993) 1511–1522.

[14] J. M. Keller, S. Chen and R. M. Crownover, Texture Description and Segmentation through Fractal Geometry, *Computer Vision, Graphics, and Image Processing* **45** (1989) 150–166.

[15] J. L. Vehel, Using Fractal and Morphological Criteria for Automatic Classification of Lung Diseases, *Proceedings of the SPIE* **1199** (November 8-10, 1989), 903–912.

[16] H. Kaneko, A Generalized Fractal Dimension and Its Application to Texture Analysis, *IEEE International Conference on Acoustics, Speech, and Signal Processing* (May 23-26, 1989) 1711–1714.

[17] T. J. Dennis and N. G. Dessipris, Fractal Modelling in Image Texture Analysis, *IEE Proceedings* **136** Part F 5 (1989) 227–235.

[18] B. E. Cooper, D. L. Chenoweth and J. E. Selvage, Fractal Error for Detecting Man-made Features in Aerial Images, *Electronics Letters* **30** 7 (March 31, 1994) 554–555.

[19] S. Pruess, *Some Remarks on the Numerical Estimation of Fractal Dimension* (Colorado School of Mines, Golden, CO, August 1990).

[20] R. F. Voss, Random Fractals: Characterization and Measurement, *Scaling Phenomena in Disordered Systems* (Plenum Press, New York, 1985) 1–11.

[21] A. Rosenfeld and A. Kak, *Digital Picture Processing* **2** (Academic Press, New York, 1982).

Pixel distances

$\Delta x = 1$
$\Delta x = \text{sqrt}(2)$
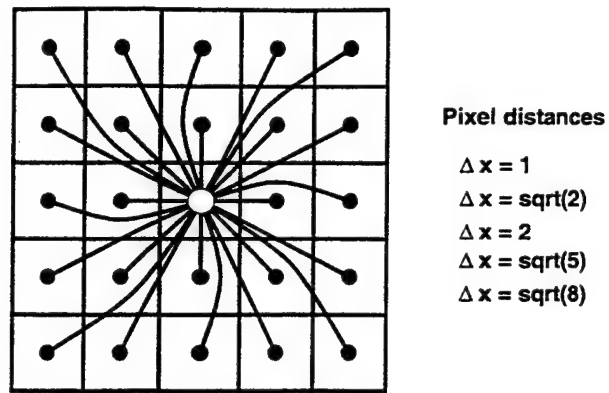$\Delta x = 2$
$\Delta x = \text{sqrt}(5)$
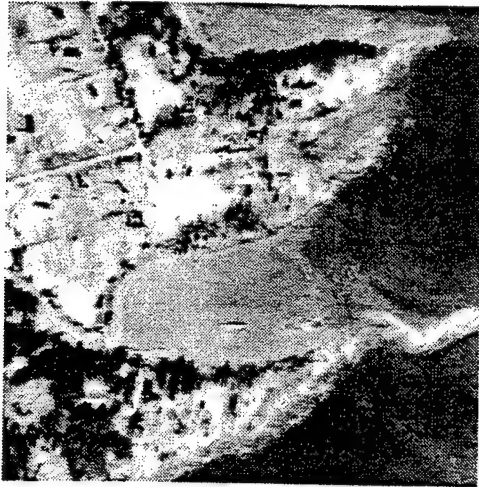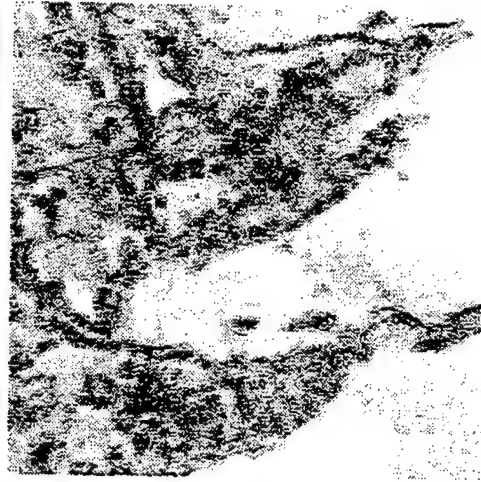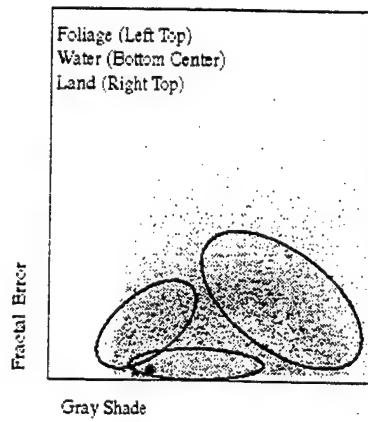$\Delta x = \text{sqrt}(8)$

Fig. 1. Pixel Spacings Used with the Center-Oriented Method

(a) Original Image

(b) Fractal Error Image, Negated

Foliage (Left Top)
Water (Bottom Center)
Land (Right Top)

Fractal Error

Gray Shade

(c) Histogram

(d) Classified Image

Fig. 2. Fractal Error and Image Intensity Applied to the Shore Scene

17

(a) Correct Segmentation   (b) Classification after Removal of
Small Regions

Fig. 3. Comparison of Classification Quality
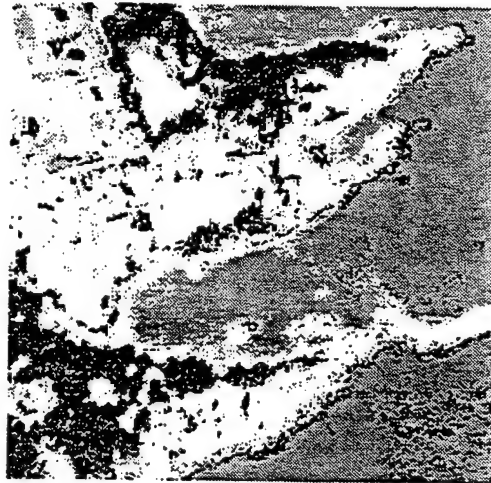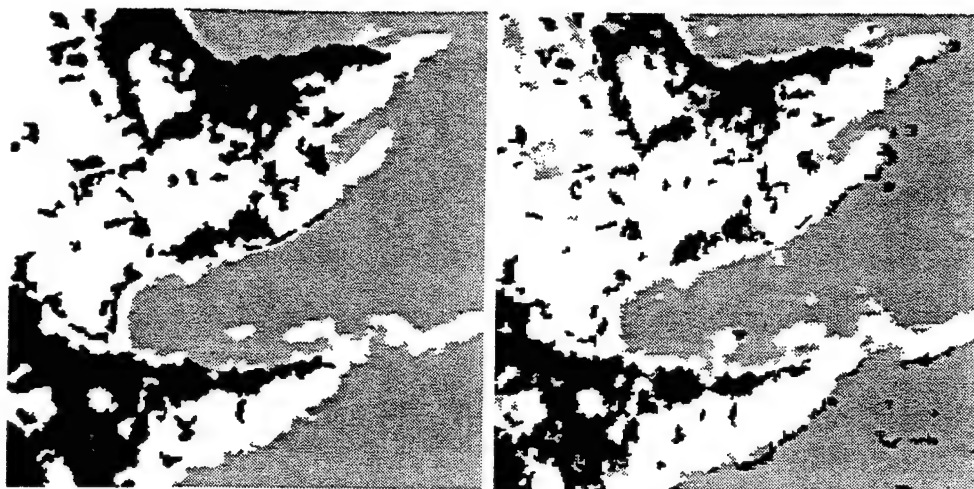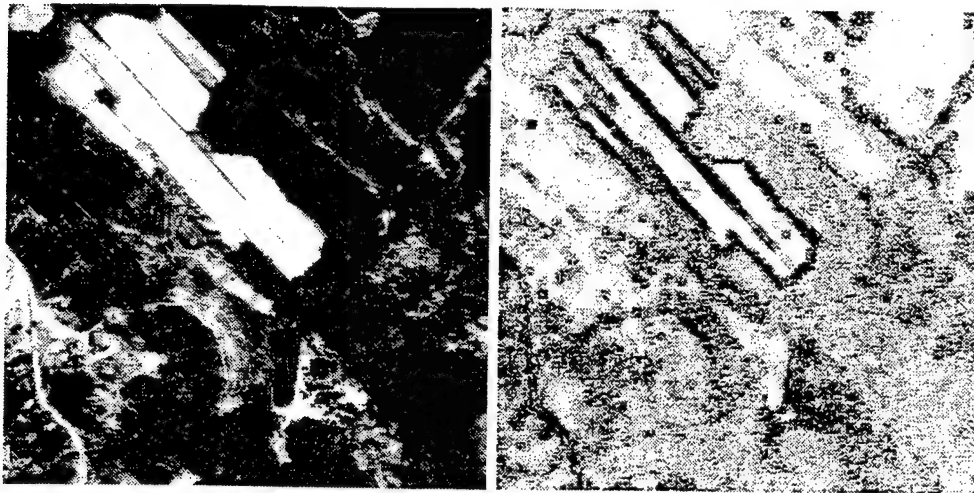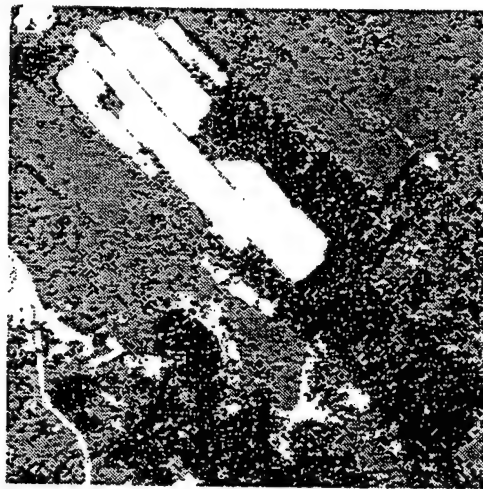
(a) Original Image

(b) Fractal Error Image, Negated

(c) Classified Image

Fig. 4. Fractal Error and Image Intensity Applied to the Farm Scene
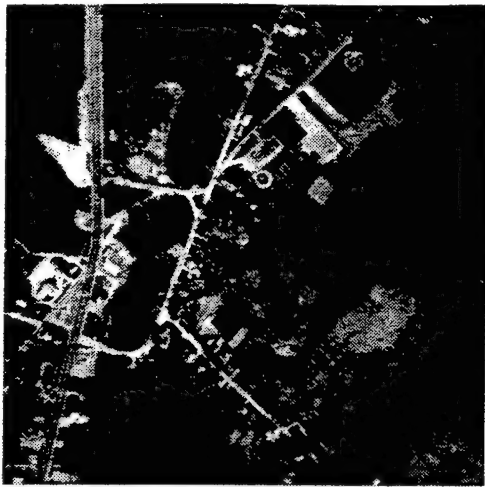
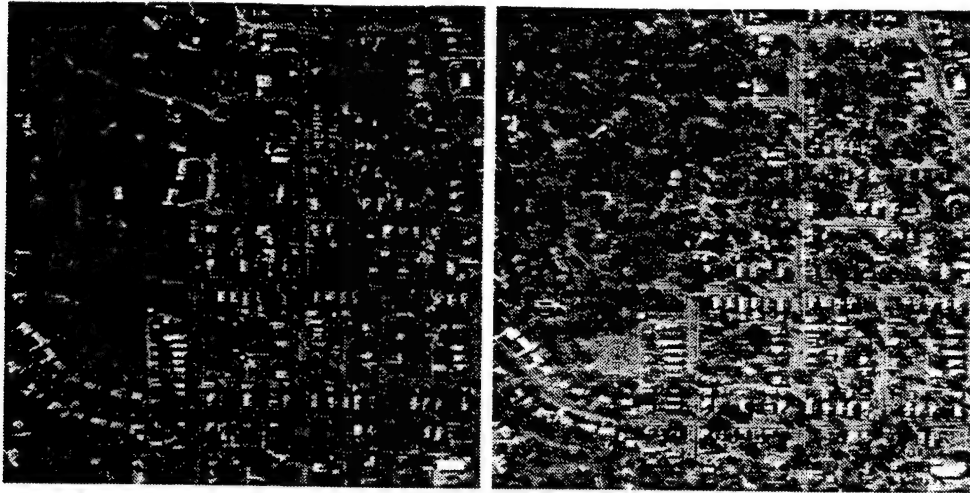(a) Original Image
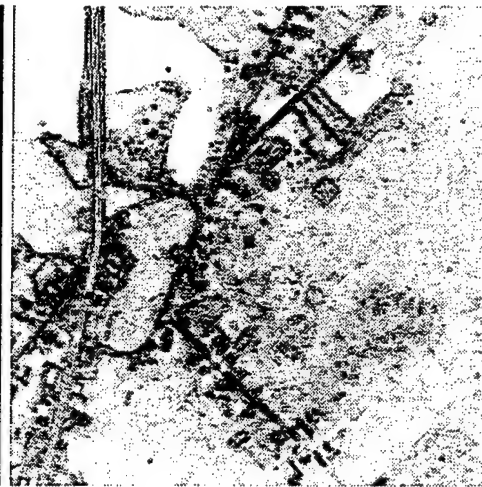
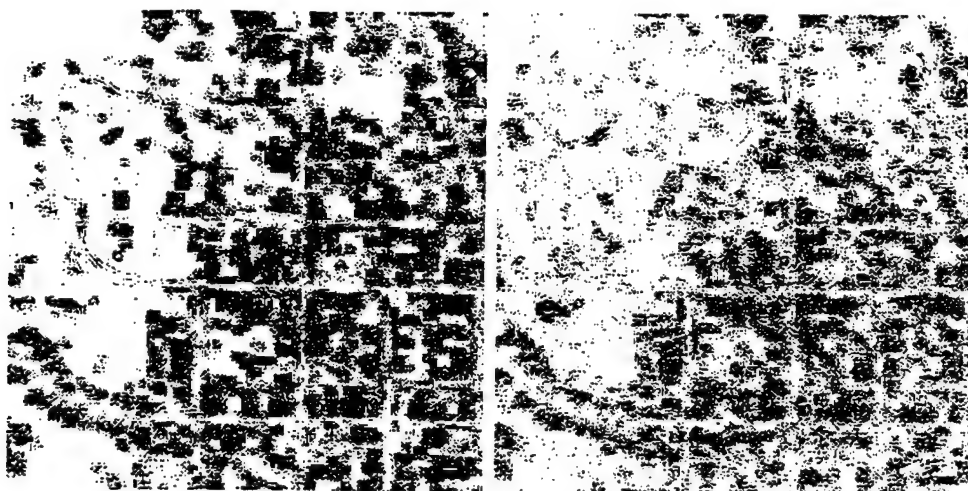(b) Fractal Error Image, Negated
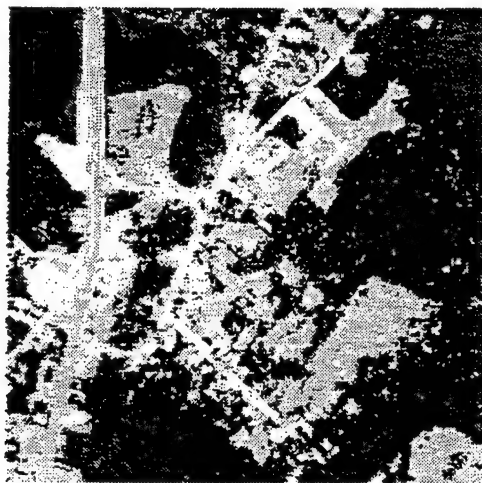
(c) Classified Image

Fig. 5. Fractal Error and Image Intensity Applied to the Mixed Scene

(a) April   (b) October

Fig. 6. Residential Scene at Two Times of the Year

(a) April   (b) October

Fig. 7. Binary Fractal Error Images of Residential Scene

Fig. 8. Correct Segmentation of Residential Scene

Biographical Sketch

Brian E. Cooper received his Ph.D. degree from the University of Louisville in 1994. His research interests include texture representation, fractal-based image modeling, halftoning, image and data compression, and approximate reasoning. He currently serves as a software research engineer with Lexmark International in Lexington, KY.

Darrel L. Chenoweth is Professor and Chairman of Electrical Engineering at the University of Louisville. He joined the University of Louisville in 1970 after completing his Ph.D. at Auburn University. He has been involved in image processing and pattern recognition research sponsored by the Naval Air Warfare Center and the Office of Naval Research since 1981. He is a Fellow in the Institution of Electrical Engineers.

E. David Jansing earned his B.S. in 1991 and his M.Eng. in 1992 from the University of Louisville. He completed his Ph.D. in Computer Science and Engineering at the same institution in 1997. He is currently an instructor at the University of Louisville in the Electrical Engineering department. His research interests include computer vision, image processing and artificial intelligence, especially in the field of genetic algorithms.

John E. Selvage is a doctoral student at the University of Louisville, where he has received B.S. and M.Eng. degrees in Computer Science and Engineering. He is a former Senior Engineer at Martin Marietta Orlando Aerospace. He has done research in image processing, particularly object detection and classification algorithms.

Two-Dimensional Entropic Segmentation Of Cultural Objects In Aerial Imagery

E. David Jansing [a], Thomas A. Albert [b], and
Darrel L. Chenoweth [a]

[a]*University of Louisville*
Speed Scientific School
Department of Electrical Engineering
Louisville, KY 40292

[b]*University of Louisville*
School of Medicine
Center for Applied Microcirculatory Research
Louisville, KY 40292

Abstract

A novel method of two-dimensional entropic segmentation using a linear discriminant function is presented. This segmentation approach highlights man-made objects against background using features that provide two separable classes. It will be shown that aerial images can be segmented accurately using fractal error and gray levels as features.

Key words: entropic segmentation, man-made object recognition, classification, remote sensing

1 Introduction

Segmentation is an important task in many image processing systems. Automatic target recognition often uses segmentation to separate the desired target from the background. Segmentation is also used to automatically highlight certain parts of the image that may be difficult to see with the human eye, such as man-made objects in synthetic aperture radar (SAR) imagery.

Thresholding intensity values is the simplest method of segmentation. This type of thresholding implies that the object or objects in question have intensities that are distinctly different from the background or other objects in the scene. e.g. a black part on a white conveyor belt.

Threshold selection methods can be grouped into two categories. local methods and global methods. Segmenting an entire image with a single value using the gray-scale histogram is an example of a global method. Local methods partition an image into a group of sub-images and select a threshold point or set of threshold points for each of the sub-images. Global thresholding techniques are easy to implement. but some methods tend to be inaccurate, especially with complex images. The reader is encouraged to reference Sahoo et al. (1988) and Wong and Sahoo (1989) for excellent surveys of global thresholding methods using information theory.

This paper presents a novel segmentation method after a technique presented by Sahoo et al. (1997). This method finds a thresholding line using the information from a two-dimensional probability density function. Sahoo et al. proposed an entropic global thresholding method that selects a threshold point based on minimizing the difference between entropies of the object and the background distributions of the probability density function. The method presented here uses this thresholding point along with statistical information from the assumed data sets of the object and the background to produce a second point. The line that is produced from the two thresholding points is then used

to separate the object data set from the background data set. The data is derived from an aerial image in which the object data set is the cultural objects present.

2   2-D Extension Of The Entropy Crossover Method

This section will outline the 2-D extension of the entropy crossover method described in Sahoo et al. (1997) as well as the novel technique for the proposed extension. The information in the gray-level frequency distribution (histogram) is often not enough to accurately segment a given image. In these cases, it is not uncommon to incorporate additional information in the histogram. The additional information results in a 2-D histogram which is the frequency distribution of the second feature combined with the gray-level frequency distribution. This 2-D histogram is an estimate of the joint probability density function of the two features. The co-occurrence matrix is a commonly used example of this type of 2-D histogram. Obviously, the choice of features is extremely important in determining the separability of the object and background classes in the joint probability density function.

Abutaleb (1989) proposed that the optimal threshold point for a 2-D histogram can be found by locating an optimal threshold for each feature in the histogram, which resulting in two separate thresholds, $t1^*$ and $t2^*$. When the threshold for each feature has been found, the two orthogonal lines segment-

3

ing each feature divide the matrix into four quadrants. The intersection of the orthogonal lines produces the overall thresholding point $(t1^*, t2^*)$, as shown in Figure 1.

Brink (1992) used two of the quadrants for segmentation. He segmented Quadrant II pixels as background and Quadrant IV pixels as foreground. However, he discarded the pixels located in Quadrants I and III, which may ignore important information concerning the objects to be segmented. Instead of thresholding in quadrants, a thresholding *line* in the 2-D histogram plane would provide better segmentation. Such a line is shown in Figure 1. Sahoo et al. recognized that the selection of a single point to classify the 2-D pdf may completely disregard some sections of the desired distribution that reside near, but not in, the foreground or background quadrants. They suggested finding the optimal thresholding line by minimizing the difference in the entropy values on either side of a proposed line in the 2-D histogram plane, not unlike the method used in finding the optimal point in the 1-D histogram. However, this is a costly method, as there are approximately 4×10^5 lines that intersect a 256x256 histogram.

Instead of an exhaustive search of the possible thresholding lines, it is possible to calculate a thresholding line using the statistics of each quadrant. Let $(\overline{x}, \overline{y})_j$ be the central moment of quadrant $j = I, \cdots, IV$. Thus.

4

Table 1
Automatic 2-D Entropic Segmentation Algorithm

1. Calculate the optimal thresholding point $(t1^*, t2^*)$ using the entropy crossover method as described in Sahoo et al. (1997).

2. Divide the 2-D histogram into quadrants around $(t1^*, t2^*)$.

3. Determine the quadrant, j, with the largest number of points and calculate its central moment, $(\overline{x}, \overline{y})_j$.

4. Compute the line, $l(x, y)$ that intersects $(t1^*, t2^*)$ and $(\overline{x}, \overline{y})_j$.

5. Define $L(x, y)$ to be the line perpendicular to $l(x, y)$, passing through $(t1^*, t2^*)$.

6. Segment the image using $L(x, y)$ as the optimal thresholding line.

$$\overline{x} = \frac{\Sigma x_i}{p_j} \tag{1}$$

$$\overline{y} = \frac{\Sigma y_i}{p_j} \tag{2}$$

where p_j is equal to the total number of points in quadrant j. The quadrant with the most points is determined, and its central moment is used with the optimal threshold point to form a line. The resulting line is perpendicular to the desired thresholding line. The algorithm is summarized in Table 1.

3 Direct Texture Measurement

One very useful feature for segmentation is image texture. While there is no formal definition of texture, there are several means by which to measure different properties of texture, such as smoothness, coarseness, and regularity (Gonzalez and Woods, 1992). These properties may be quantified using sta-

5

tistical. structural. spectral. or fractal methods. An example of a statistical texture measure is standard deviation, which quanitifies similarity of pixels within a localized neighborhood (Albert et al.. 1991). Cooper et al. (1994) described a fractal analysis approach to texture called "fractal error". This metric is based on the observation that natural features in high-resolution gray scale aerial images fit the fractional Brownian motion (fBm) mathematical model. as described by Mandelbrot (1977).

Cooper (1994) used gray-scale values and fractal error as features in a 2-D statistical classifier to successfully detect different objects in aerial images. Jansing et al. (1997) demonstrated that fractal error was useful in identification of cultural objects in synthetic aperture radar (SAR) imagery. Because of its proved ability to differentiate between textures. fractal error along with the pixel gray level value were used as features in the segmentation algorithm.

4   Results

The 2-D histogram and calculated threshold line are illustrated in Figure 3. The result of applying the 2-D entropic segmentation method is shown in Figure 4. The cultural objects are highlighted and the natural terrain is classified as background. Figure 5 shows ground truth for the suburban aerial image. Table 2 shows the comparison of the 1-D segmentation errors using intensity and fractal error as independent features. with the error values for the

6

Table 2
Results of 1-D and 2-D Segmentation Using Intensity and Fractal Error As Features

Segmentation Feature(s)	Type I Error	Type II Error	Total Error
Intensity	15 %	10 %	25 %
Fractal Error	0 %	81 %	81 %
Intensity & Fractal Error	4 %	12 %	16 %

proposed 2-D segmentation algorithm. These features were found to be highly uncorrelated, with an r-value of 0.18, and therefore were well-suited for this task. It is clear that the 2-D method provides a more accurate segmentation.

The Type II errors may be due, in part, to the occlusion of houses and other cultural objects by the trees in the image. Also, the ground truth did not include the streets visible in the image, however, these cultural objects were also correctly segmented using the method. Therefore, the results of classification of man-made objects was actually better than 16 % for the 2-D entropic segmentation.

5   Conclusions

A novel method of two-dimensional entropic segmentation was presented. While the results for an aerial image are accurate, it is clear that success with real-world images is dependent on an intelligent choice of features. Namely, features should be uncorrelated to provide the best separability of classes in the joint probability density function. For aerial images, direct texture measures such as fractal error often provide a good feature for segmentation. Future

7

research should focus on a study of segmentation with different features. Such a study may provide an indication of which features may be more effective in highlighting cultural objects in aerial imagery using this entropic segmentation method.

References

Abutaleb, A. S. (1989). Automatic thresholding of gray-level pictures using two- dimensional entropy. *Computer Vision, Graphics and Image Processing*, 47, 22-32.

Albert, T. A., O'Conner, C. A., and Harris, P. D. (1991). Automatic segmentation of microvessels using textural analysis. In *Proceedings of SPIE*, volume 1450, 84-89.

Brink, A. D. (1992). Thresholding of digital images using two-dimensional entropies. *Pattern Recognition*, 25, 8, 803-808.

Cooper, B. E. (1994). *Fractional Brownian Motion For Representation Of Natural Image Texture*. Ph.D. thesis, University of Louisville.

Cooper, B. E., Chenoweth, D. L., and Selvage, J. E. (1994). Fractal error for detecting man-made features in aerial images. *Electronics Letters*, 30, 7, 554-555.

Gonzalez, R. C. and Woods, R. E. (1992). *Digital Image Processing*. Addison-Wesley, Reading.

Jansing, E. D., Chenoweth, D. L., and Knecht, J. (1997). Feature detection

in synthetic aperture radar images using fractal error. In *Proceedings of the IEEE Aerospace Conference*, volume 1, 187–195.

Mandelbrot, B. B. (1977). *Fractals: Form, Chance and Dimension*. W. H. Freeman and Co., San Francisco.

Sahoo, P. K., Slaaf, D. W., and Albert, T. A. (1997). Threshold selection using a minimal histogram entropy difference. *Optical Engineering*, 36, 7, 1976–1981.

Sahoo, P. K., Soltani, S., Wong, A. K. C., and Chen, Y. C. (1988). A survey of thresholding techniques. *Computer Vision, Graphics and Image Processing*, 41, 233–260.

Wong, A. K. C. and Sahoo, P. K. (1989). A gray-level threshold selection method based on maximum entropy principle. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-19, 866–871.

Figure Captions

Figure 1 – The optimum thresholding line for the 2-D histogram shown.

Figure 2   Aerial image of suburban area.

Figure 3   2-D frequency distribution showing resulting threshold line.

Figure 4   Aerial image of suburban area segmented using gray levels and fractal error as features.
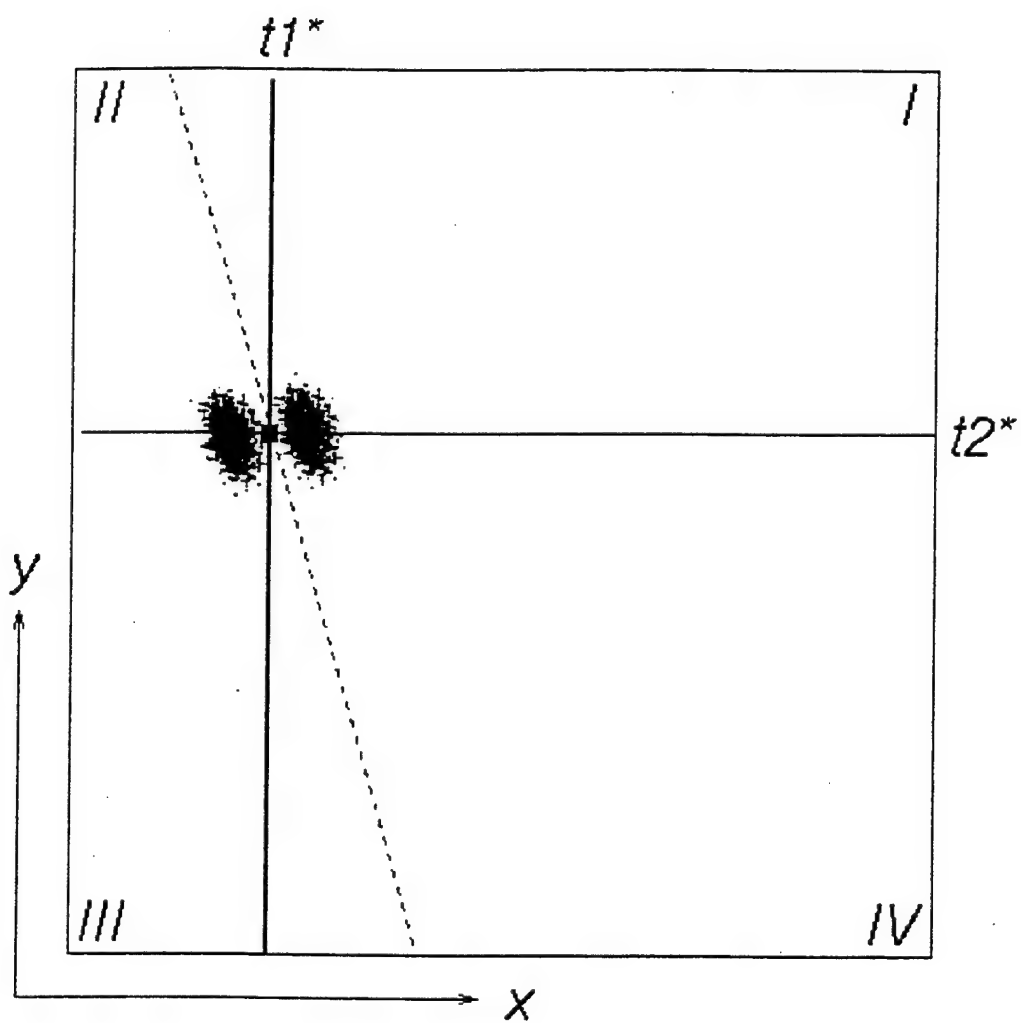
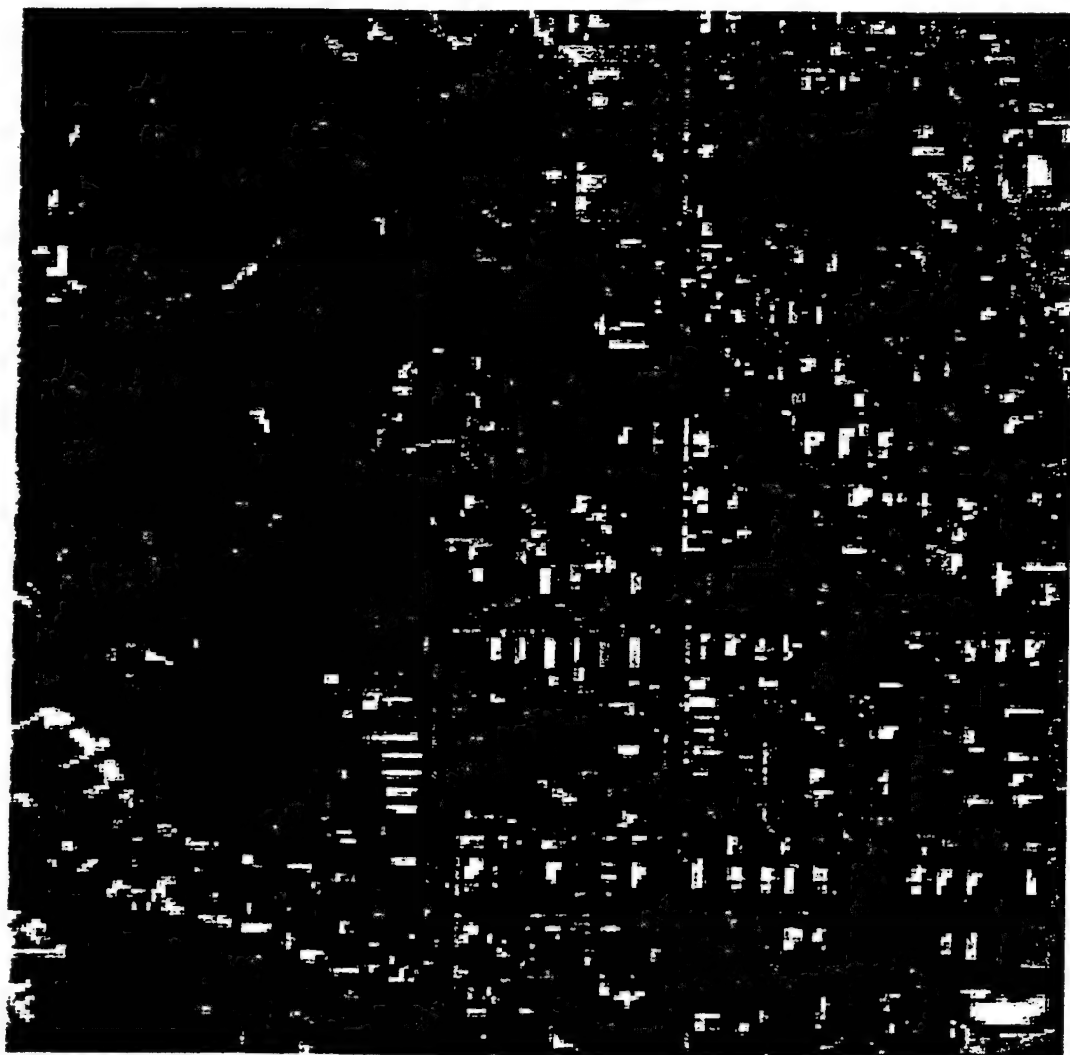Figure 5 – Ground truth for the suburban aerial image shown in Figure 4.
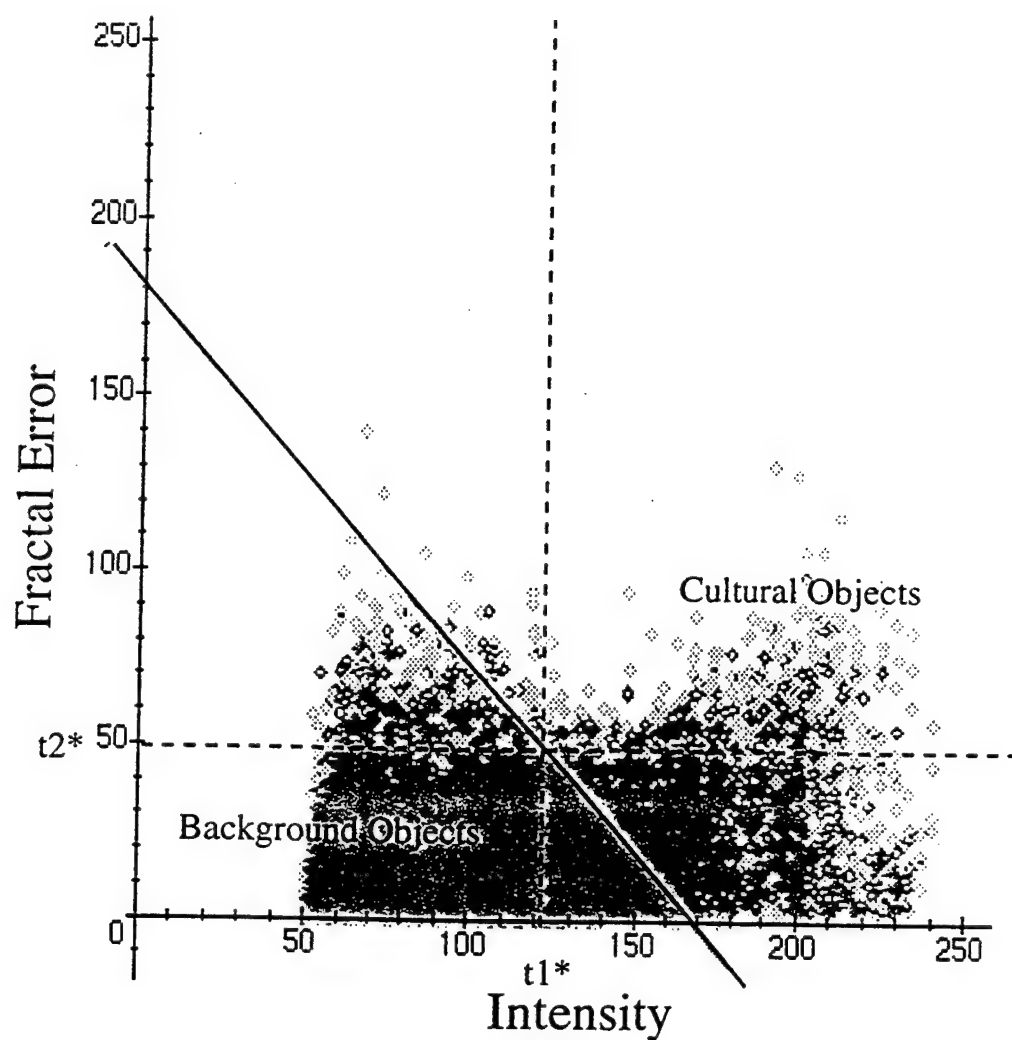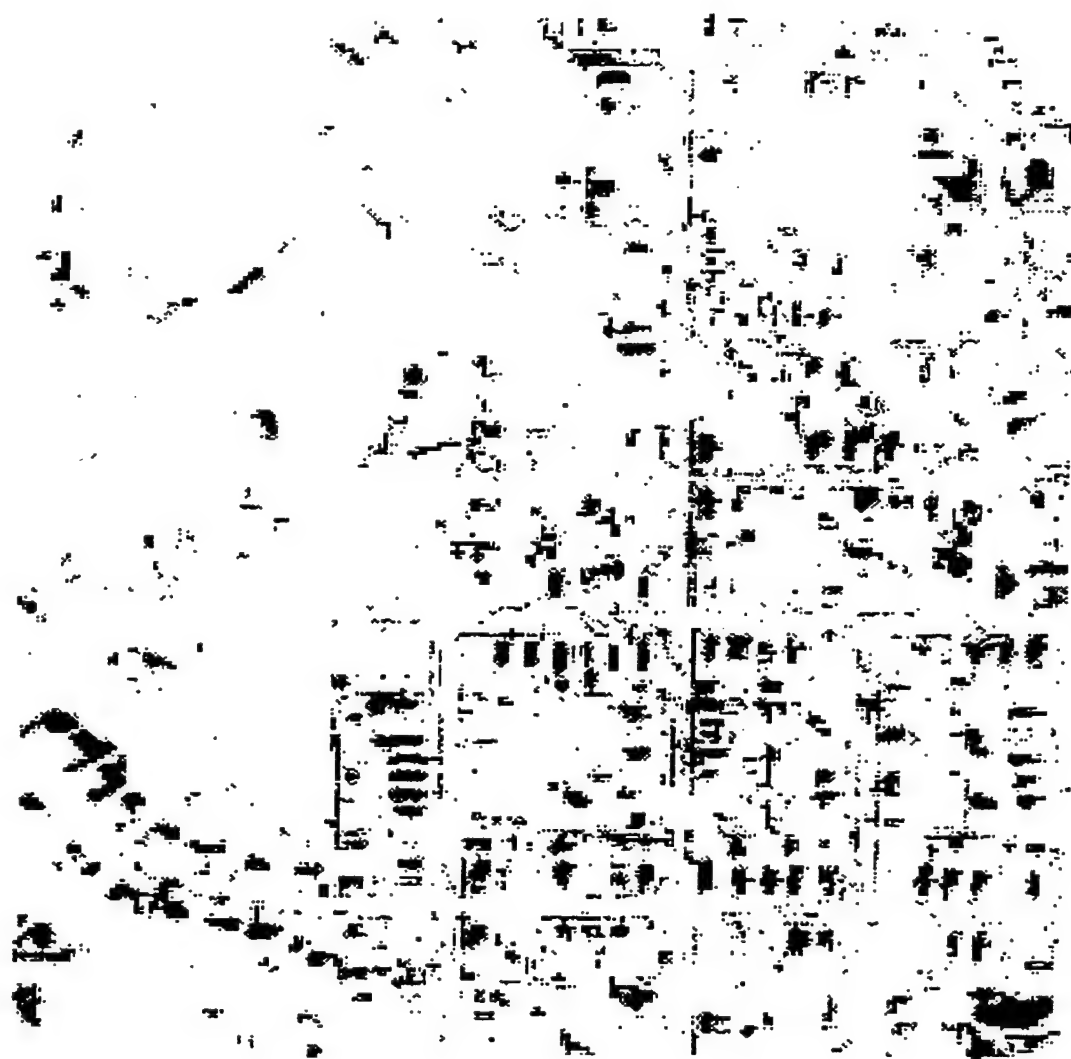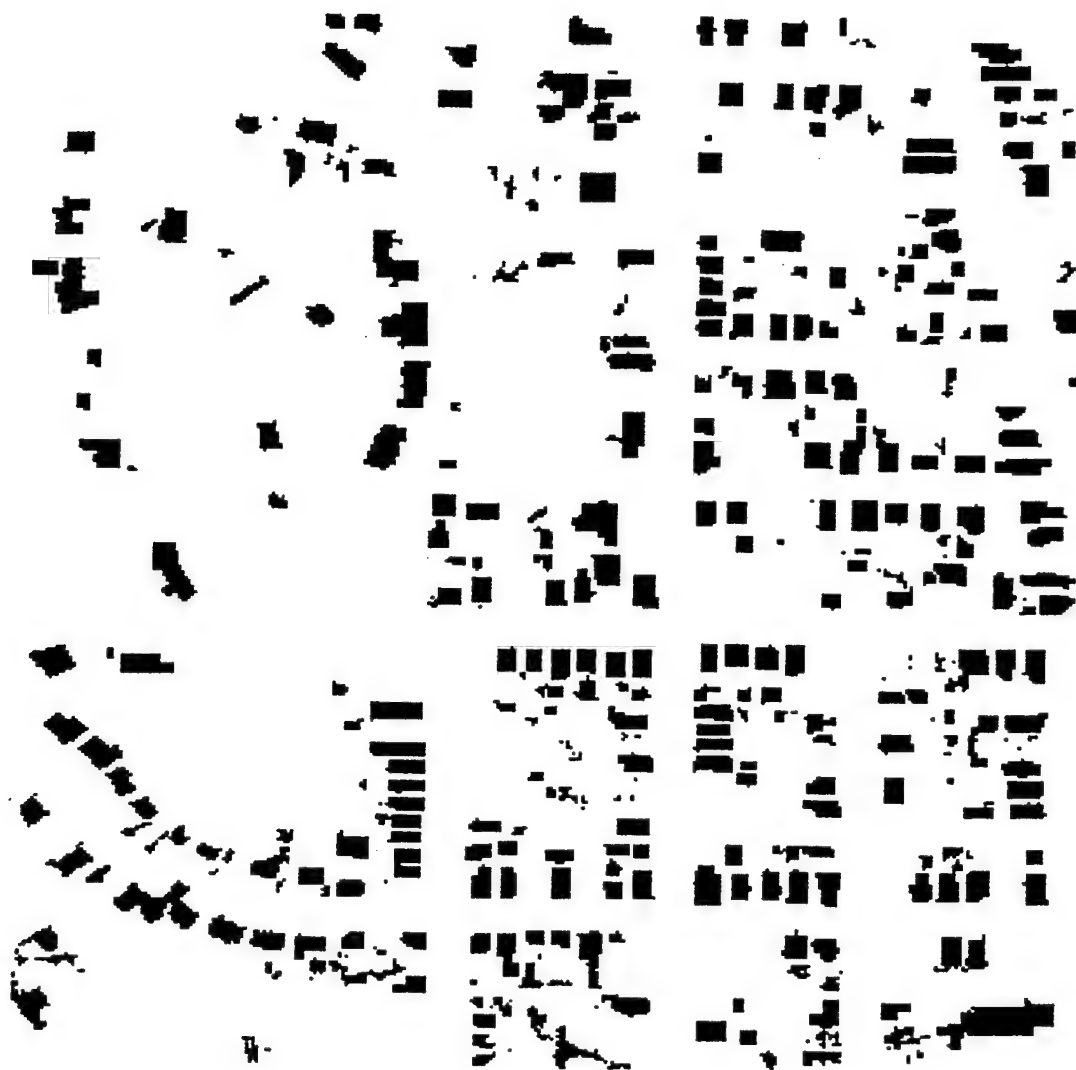
Fig. 1.

Fig. 2.

Fig. 3.

Fig. 4.

14

Fig. 5.

15

A Genetic Approximation Of A Fractal-Based Measure

E. David Jansing and Darrel L. Chenoweth
Department of Electrical Engineering
University of Louisville
Louisville, KY 40292
502-852-6289
edjans01@starbase.spd.louisville.edu
dlchen01@starbase.spd.louisville.edu

November 24, 1997

Abstract

Fractal error is an image processing metric that can be used to locate man-made features in aerial images. The metric can aid photointerpreters in locating targets in aerial reconnaissance images. Fractal error was developed for this purpose by Cooper et al. [1]. Since the development, Jansing et al. [2] have shown that the fractal error metric also works well for extracting features in synthetic aperture radar (SAR) images. Jansing et al. [3] have also shown that the fractal error metric is useful for locating edge pixels in industrial images. The fractal error metric has a wide range of applications; however, some applications require real-time image analysis. The main disadvantage of the fractal error algorithm is that it can take several seconds to compute on large images. Therefore, it is desirable to create an approximation of fractal error to provide real-time image analysis. This paper presents a novel approximation of fractal error using a genetic algorithm.

The results obtained using the approximation are compared with those obtained from the fractal error algorithm. The genetic algorithm does preserve all of the desired features of the original fractal error image, while providing a faster computation time.

Future work will consist of developing a genetic programming technique to evolve a mathematical structure for a better fractal error approximation.

1 Introduction

The mathematical nature of textures within images has generated much interest in the literature. Fractal dimension and lacunarity are often the focus of such interest. Notably, Pentland [4] used fractal dimension to analyze natural visual textures. However, little attention has been given to measuring the fitness of the fractal model itself. There has also been other work that has attempted to apply fractal characteristics and measures to basic image analysis, such as texture segmentation [5], [6]. Using a fractional Brownian motion (fBm) model, Stewart et al. [7] demonstrated the application of fractal random process models as features in the analysis and segmentation of SAR imagery. The fractal dimension of natural textures, such as grass and trees, was computed and used as texture features in a Bayesian classifier.

Stewart discussed how metrics described by fractal geometry provide accurate measures of "roughness" and "irregularity" in scale-invariant natural forms. Moghaddam et al. [8], [9] have used the local fractal dimension, a fractal metric, for segmentation and analysis of infrared (IR) imagery. Likewise, Cooper [10] implemented a localized version of the fractal error measurement as well as other fractal metrics for the segmentation of aerial imagery. Stein [11] introduced a slightly different fractal error metric, calculated by the covering method similar to the method suggested by Peleg [6]. Solka, Rogers and Priebe [12] introduced a power law signature similar to Stein's. There were significant differences, however. Stein designed a discrimination scheme employing the slope and standard error of fit of the regression line, not unlike the algorithm Cooper implemented, which is outlined in this work. Solka, Rogers and Priebe used different features to estimate fractal error. Also, Stein's decision rules were heuristic in nature, whereas Solka, Rogers

and Priebe proposed advanced density estimation techniques in an effort to fully characterize the decision surfaces which originate from their power law signatures.

In another paper, Rogers, Solka and Priebe [13] outlined a method to calculate fractal dimension using a parallel distributed processing (PDP) approach. *A priori* boundary information was incorporated into their covering method, improving their segmentation results.

Cooper et al. [1] developed a fractal error metric based on the observed propensity of natural image features to fit an fBm model. They used this feature in a statistical classifier to successfully segment regions in aerial reconnaissance images. Jansing et al. [2] also used this same feature to segment man-made objects in SAR imagery.

Upon examination, there are two important examples that provide motivation for fractal error analysis. The first example involves segmentation of cultural objects in SAR imagery, and the second is reliable edge detection in the presence of noise.

As an illustration of the need for segmentation, we will examine mission planning, a critical part of any tactical aircraft mission. To ensure the safety of the pilot and the success of the mission, extensive planning is required. Greater efficiency and consistency are achieved by supplementing pilot calculations with machine automation and intelligence. However, during the actual execution of the mission, the tactical situation may undergo unpredictable change. Instead of returning to the aircraft carrier and re-planning a new mission, it may be desirable to modify the mission en route. Therefore, it is necessary to data link mission-related information to aid the pilot with his new objective. Surveillance and reconnaissance imagery coupled with other intelligence information and digital maps can be presented to the flight crew. Included in this mission-related data may be processed imagery of the target area from an off-board sensor. These off-board sensors may provide IR imagery, SAR imagery, and optical imagery. SAR imagery is a likely sensor candidate. While highly specular and often difficult to interpret without enhancement, SAR is less susceptible to weather conditions and ambient illumination than IR. Thus, from all of the data sent to the flight crew, terrain features, line-of-sight perspectives, and expected target area features can be displayed in order to highlight the new mission aspects. The fractal error metric is described here in a method that provides useful detection of cultural objects. The resulting imagery could be used in on-the-fly mission adjustments.

As another example, edges typically do not fit the fractional Brownian motion (fBm) model well, due to their irregularity. Therefore, fractal error is also a practical edge enhancement operator. Jansing et al. [3] developed an algorithm utilizing fractal error to separate the edges from the other features in images.

This paper presents an approximation of Cooper's [10] fractal error measure using a genetic algorithm. Computational expense is the unfortunate by-product of Cooper's algorithm. Since many applications require real-time computation to be practical, it is desirable to create an approximation of fractal error to provide real-time analysis, whether for the detection of cultural objects or edges.

2 Fractal Error

2.1 Definition Of A Fractal

It is well known that many textures and scenes can be modelled as *fractals*. A fractal, according to La Brecque [14], "has a rough shape to one degree or another made of parts which, when magnified, resemble the whole." It is also well known that literature describing fractals often lacks precision when attempting to define what a fractal is. However, the reader of fractal geometry and theory can turn to Falconer [15] for a detailed description of the properties of fractals.

Definition 1 *The set F is a fractal, if it has the following properties:*

1. *F has a fine structure, that is, detail on arbitrarily small scales.*

2. *F is too irregular to be described in traditional geometrical language, both locally and globally.*

3. *F often has some form of self-similarity, perhaps approximate or statistical.*

4. *Usually, the "fractal dimension" of F (defined in some way, and there are several unique definitions) is greater than its topological dimension.*
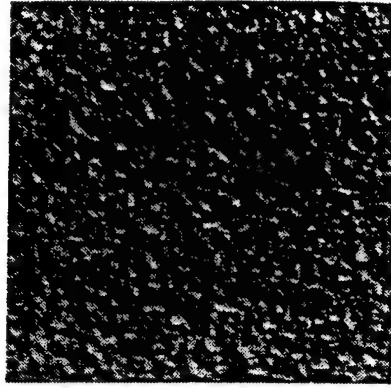
Figure 1: Texture image of cork.

5. *In most cases of interest. F is defined in a very simple way. perhaps recursively (e.g.. the Julia or Mandelbrot Sets).*

2.2 Fractal Dimension

How are fractals distinguished between one another? How does one measure the size of fractals? What measure can be used to compare and contrast fractals?

Fractal dimension is the measure that is generally used to distinguish between fractals, giving the fractals a measurement of "size". This numeric representation attempts to quantify a subjective quality which one might have about how densely the fractal occupies the space in which it exists.

Fractal dimension is just as difficult to define as fractals themselves. Mandelbrot [16], Falconer [15], Peitgen et al. [17] and Edgar [18] provide excellent discussions of many different fractal dimension definitions. Each fractal dimension definition has a distinct style. Although the definitions are all related, Peitgen [17] claims that some definitions make sense in certain cases, while other definitions may not be appropriate in the same case. Experience and heuristics prompt the selection of an appropriate fractal dimension definition, according to the application.

2.3 Lacunarity

Mandelbrot [16] defined another measure for fractals. *Lacunarity* describes the "holiness" ([19], pg. 236) of an occupied fractal lattice. The origin of the name lacunarity can be appreciated by looking at an image of cork in Figure 1. This image is part of a collection of texture images, presented by Brodatz [20]. From the Latin word "lacona," which means gap. lacunarity represents the gaps within a fractal structure. Thus, the percentage of spaces between the cork in Figure 1 is the measure of lacunarity. Practically. lakes or other natural objects within aerial images may be classified by using lacunarity as a feature measure.

2.4 Fractional Brownian Motion

Fractals may occur in many different forms. Mandelbrot [21] was the first to define *fractional Brownian motion (fBm)*. Brownian motion refers to the erratic motion of small suspended particles. resulting from random collisions with other particles. Fractional Brownian motion is an extension of this model. Cooper [10] gives an excellent description of fBm; it will be summarized here.

The function of fBm is defined as the differences between successive samples. Let $B_H(t)$ represent a fBm signal. where t is a vector containing E independent variables. Then the increment of the fBm signal is described as $\Delta B_H = B_H(t_2) - B_H(t_1)$. where t_1 and t_2 are two distinct points in time. The measure ΔB_H is normally distributed with a mean of zero and a variance proportional to the $2H$ power of the Euclidean distance. The mean takes the form of

$$E[B_H(t_2) - B_H(t_1)] = 0. \tag{1}$$

Likewise, the variance is defined as

$$Var[B_H(t_2) - B_H(t_1)] = \sigma^2 |t_2 - t_1|^{2H}. \tag{2}$$

where σ^2 is the proportionality constant of the variance and H is the Hurst parameter, which must be strictly $0 < H < 1$. Note that when $H = 0.5$, the fractional Brownian motion model is equivalent to the classical Brownian motion model. The value of H can be used to describe the fractal dimension D as

$$D = E + 1 - H. \tag{3}$$

where E is the Euclidean dimension (or, the number of independent variables of $\mathbf{t}$). It is therefore easy to see that small values of H produce high fractal dimension and large values of H produce a low fractal dimension. Combining the equations for mean and variance, while also taking into account the fractal dimension, we can arrive at the following relationship:

$$E[|B_H(t_2) - B_H(t_1)|] = k|t_2 - t_1|^H. \tag{4}$$

The above equation is the fundamental basis for the fractal error metric. The regions observed in an actual image may be a combination of many different textures. Each texture can be represented by its fractal dimension. However, before attempting to determine the fractal dimension, it is useful to know how well a region (or window) may fit the fractal model. Thus, measuring the error produced when estimating the fractal dimension will give a useful metric in order to determine the "fractalness" of a region in the image. A small error will indicate that a region fits the fractal model well, and thus can be considered fractal. Conversely, a large error will indicate that the region fits poorly into the fractal model and thus is probably not fractal and the fractal dimension measure is useless. Mathematically, this can be defined as

$$E[|G[\mathbf{x_2}] - G[\mathbf{x_1}]|] = k|\mathbf{x_2} - \mathbf{x_1}|^H \tag{5}$$
$$E[|\Delta G_{|\Delta \mathbf{x}|}|] = k|\Delta \mathbf{x}|^H \tag{6}$$

where G is the region or window in the image and $\mathbf{x}$ is the measured distances within the region. Estimates of H and k can be found by using a linear regression scheme.

$$\ln E[|\Delta G_{|\Delta \mathbf{x}|}|] = \ln k + H \ln |\Delta \mathbf{x}|. \tag{7}$$

These estimates, $\overline{H}$ and $\overline{k}$ can then be used to calculate the error with the following equation:

$$error_{|\Delta \mathbf{x}|} = E[|\Delta G_{|\Delta \mathbf{x}|}|] - \overline{k}|\Delta \mathbf{x}|^{\overline{H}}. \tag{8}$$

Using a "center-oriented" window (i.e., a square window of $N \times N$, where N is strictly odd), there will be five, nine, or fourteen error values, given the window is 5x5, 7x7, or 9x9 respectively. Thus, a cumulative error for the model can be given by the root mean square error

$$RMS\ Error = \sqrt{\frac{1}{n} \sum_{|\Delta \mathbf{x}|} (error_{|\Delta \mathbf{x}|})^2}. \tag{9}$$

Thus, using the RMS error, it is easily determined whether or not a pixel with a surrounding 5x5, 7x7 or 9x9 region is fractal in nature.

3 Fractal Error Algorithm

Using the method described in the previous section, Cooper developed an algorithm to calculate the fractal error for each pixel in a scene. This algorithm is described in detail in Table 1.

The following will outline the steps of the entire fractal error algorithm. Note that this example will produce a single number that will represent the error for the center pixel in relation to its neighbors. Figure 2 represents a sample 5x5 window. Since the localized neighborhood is 5x5, there are five unique distances in the window, as shown in Figure 3.

Table 1: Fractal Error Algorithm

1. Define a 5x5, 7x7, or 9x9 sliding window.
2. Calculate Δx and $E[|\Delta G|]$ for each pixel in the neighborhood of the sliding window.
3. Using linear regression, find the slope and the y-intercept for each unique Δx in the window from the equation $\ln(E[|\Delta G_{|\Delta x_i|}|]) = \ln(k) + H|\Delta x_i|$.
4. Derive $\overline{H}$ = slope and $\overline{k}$ = exp(y-intercept) from the above relationship.
5. Using $error_{|\Delta x_i|} = E[|\Delta G_{|\Delta x_i|}|] - \overline{k}|\Delta x_i|^{\overline{H}}$, calculate the fractal error for each unique Δx.
6. Compute RMS error by $RMS\ Error = \sqrt{\frac{1}{n}\sum_{|\Delta x_i|}(error_{|\Delta x_i|})^2}$.
7. Save RMS error for that pixel, move the window, and repeat the process over the entire scene.

250	200	220	200	200
175	210	170	159	100
110	100	120	115	100
96	200	205	210	211
95	201	197	205	200

Figure 2: A sample 5x5 window.

2.83	2.23	2.00	2.23	2.83
2.23	1.41	1.00	1.41	2.23
2.00	1.00		1.00	2.00
2.23	1.41	1.00	1.41	2.23
2.83	2.23	2.00	2.23	2.83

Figure 3: The Euclidean distances over the 5x5 neighborhood.

Table 2: Distances From The Center Pixel (Δx) And The Expected Value Of The Absolute Difference In Gray Scale Relevant To The Center Pixel ($E[|\Delta G|]$)

Δx	$\ln \Delta x$	$E[|\Delta G|]$	$\ln E[|\Delta G|]$
1.000	0.000	40.00	3.69
1.414	0.347	74.75	4.31
2.000	0.693	51.75	3.95
2.236	0.805	91.50	4.51
2.828	1.040	78.75	4.37

Table 3: Calculated Error From Fractional Brownian Motion Model

Δx	error
1.00	-6.06
1.41	18.4
2.00	-17.3
2.24	17.7
2.83	-5.9

Table 2 represents the absolute values of the differences of the gray scales over the unique sets of distances in the neighborhood. Thus, the gray scale value of each pixel that has a distance of 1.41 is subtracted from the gray scale value of the center pixel. Their absolute values are averaged to give $E[|\Delta G|]$ for each unique set of distances.

Using Equation 7, we can obtain estimates for the Hurst parameter and the proportionality constant, H and k respectively. These estimates can be found using linear regression [22]. If the linear model takes the form

$$y = \beta_0 + \beta_1 x, \tag{10}$$

then estimates for β_1 (slope) and β_0 (y-intercept) are defined as

$$\widehat{\beta_1} = \frac{\Sigma(x_i - \overline{x})y_i}{\Sigma(x_i - \overline{x})^2} \tag{11}$$

$$\widehat{\beta_0} = \overline{y} - \widehat{\beta_1}\overline{x}, \tag{12}$$

where $\overline{x}$ is the sample mean of x and $\overline{y}$ is the sample mean of y. In our particular case, x represents the "$\ln \Delta x$" term in Equation 7 and y represents the "$\ln E[|\Delta G|]$" term in the same equation. It easy to see that $\overline{x} = 0.577$ and $\overline{y} = 4.17$. The estimates of β_0 and β_1 can thus be calculated.

$$\widehat{\beta_1} = \frac{\sum_{i=0}^{5}(x_i - 0.577)y_i}{\sum_{i=0}^{5}(x_i - 0.577)^2} \tag{13}$$

$$= 0.585 \tag{14}$$

$$\widehat{\beta_0} = 4.17 - 0.577\widehat{\beta_1} = 3.83$$

The Hurst parameter estimate, $\overline{H}$, is equivalent to the slope of the linear model, that is, $\overline{H} = \widehat{\beta_1}$. The proportionality constant estimate, $\overline{k}$, is equivalent to the y-intercept of the linear model. $\overline{k} = \exp(\widehat{\beta_0})$. Therefore, $\overline{H} = 0.585$ and $\overline{k} = 46.1$.

Errors with respect to each unique distance set can be calculated with Equation 8. Table 3 shows the result of using Equation 8 in this example.

The overall RMS error, as defined by Equation 9, is 14.31. This number represents the "fractalness" of the center pixel relative to its neighbors. Decisions regarding the fractalness of the pixel are typically made in reference to the entire image. Thus, if the range of fractal errors is from 0 to 15, this pixel is not likely to be fractal in nature.

It is not difficult to see that the majority of computation comes from using the linear regression and calculating the errors from the estimated H and k. Thus, an approximation has been developed to improve computation time without sacrificing accuracy.

4   Genetic Approximation Of Fractal Error

To generate an approximation of fractal error, a mathematical structure is needed. Much of the information in calculating fractal error resides in the expected value of the absolute differences of the gray scales from the center pixel, grouped in uniform distances from the center pixel. This is expressed mathematically as $E[|\Delta G_{|\Delta r|}|]$. Weights are assigned to each of these expected values and all of the weighted expected values are added together to form the approximation of fractal error. This is expressed as

$$\widehat{FE} = a_0 \ln(E[|\Delta G_{\Delta r=2.82}|]) + a_1 \ln(E[|\Delta G_{\Delta r=2.23}|]) + a_2 \ln(E[|\Delta G_{\Delta r=2.00}|]) \tag{15}$$
$$+ a_3 \ln(E[|\Delta G_{\Delta r=1.41}|]) + a_4 \ln(E[|\Delta G_{\Delta r=1.00}|]).$$

The genetic approximation of fractal error (GAFE) evolves the weights $a_0, a_1, \ldots, a_4$ to try to estimate fractal error. The natural log of the expected values is used to maintain the linearity of the measure. As commonly used in the design of a GA, the probability for crossover is set at 0.7, the probability for mutation is set at 0.1, the probability for copy is set at 0.1, and the probability for migration is also set at 0.1. Migration, originally introduced by Potts et. al. [23] in 1994, helps combat the problem of premature convergence. Potts et al. allowed for multiple sets of individuals to evolve using traditional genetic operators such as crossover and mutation. Each set of individuals acts as an independent population, like a tribe or village. The migration operator chooses individuals at random and moves them to another population. The interaction prevents inbreeding and promotes reproduction among those individuals that have good characteristics, that is, high fitness values.

The migration operator used in this work is an extension of the method proposed by Potts et. al. [23]. Migration is implemented here by randomly generating a new individual and placing it in the new population. This randomness may allow for movement to a search space that the individuals in the previous population were unable to reach. This modification allows for the property of migration without the overhead of mulitple populations.

Each operator is chosen randomly from a uniform distribution. The population size and the maximum number of generations allowed for each simulation is variable. Fitness for each individual is determined from signal-to-noise ratio (SNR). SNR measures the ratio of original signal to noise introduced into a given test image. For our application, SNR is defined as

$$SNR = \frac{\sum_{r=0}^{M} \sum_{c=0}^{N} I_m(r,c)^2}{\sum_{r=0}^{M} \sum_{c=0}^{N} (I_m(r,c) - I_o(r,c))^2} \tag{16}$$

where $I_m(r,c)$ is the measured image and $I_o(r,c)$ is the original signal for MxN sized images. The lower the value of SNR, the noisier the signal is. Thus, to evolve weights with a genetic algorithm, it is desirable to maximize the signal-to-noise ratio.

The GAFE is also written in parallel to ensure fast simulation times. The evaluation of fitness for each individual in the population consumes most of the run time for the GA itself. To parallelize the algorithm, each individual is evaluated for fitness separately, running on as many CPU's as are available, as shown in Figure 4. The parallel algorithm was executed on an SGI Onyx R10000 high-performance computer, having 16 CPU's.

5   Results

Table 4 shows several different trials of GAFE and their results. The varying population size and maximum number of generations allowed is also given. The training image for each of these simulations is shown in Figure 5. Figure 6 illustrates the resulting image from each trial.

Each gene is comprised of 100 bits, 20 bits per weight. The first 19 bits are the magnitude, starting at 2^2 and ending at 2^{-16}. The 20th bit for each weight is a sign bit, allowing for positive and negative weights.
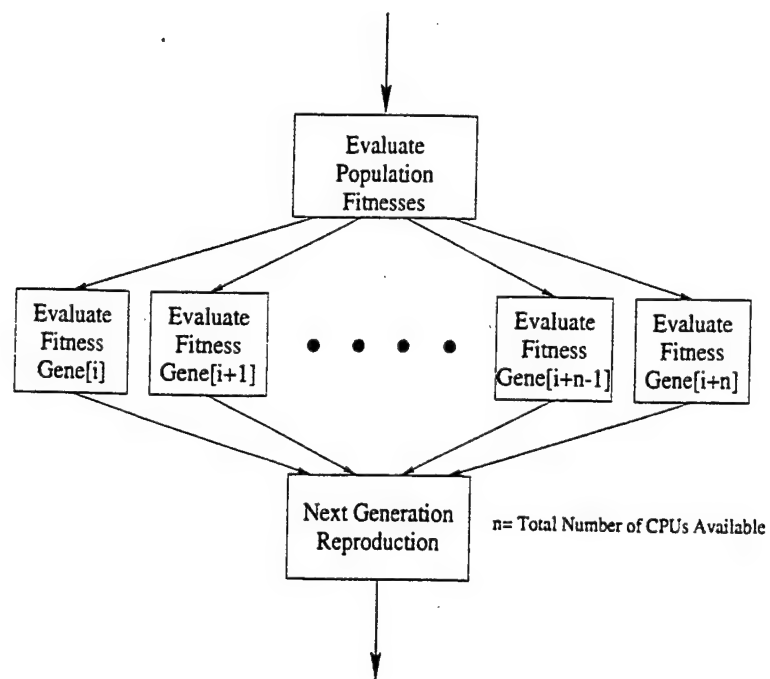
Figure 4: Parallel implementation of GAFE.

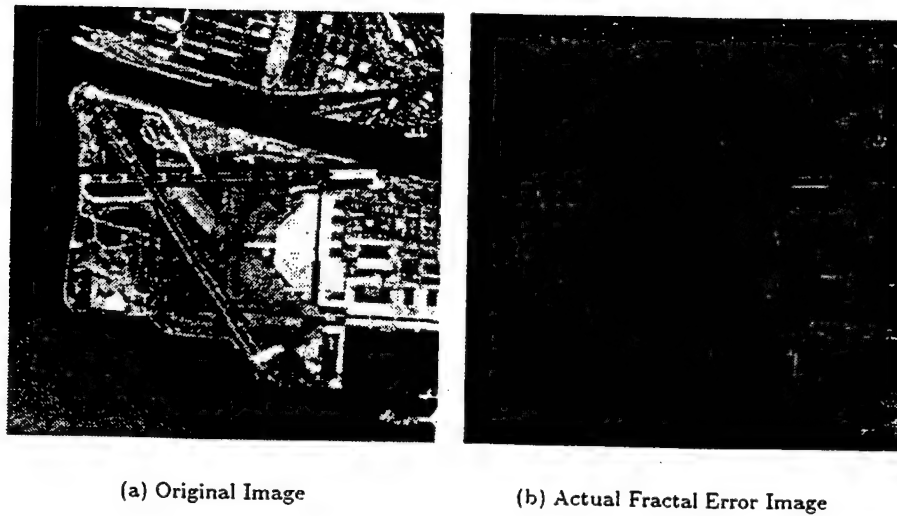
(a) Original Image
(b) Actual Fractal Error Image

Figure 5: Training image for the genetic approximation to fractal error (GAFE).

Table 4: Several Trials Of The GAFE With Varying Population Size And Generations

Trial	Pop	Max Gen	SNR
A	100	10.000	1.27
B	500	100	1.52
C	500	2000	1.56
D	1000	2000	1.63
E	2000	250	1.74
F	10.000	90	3.82

Table 5: Final Weights From GAFE, Trial F

$a_0 =$	0.0022583
$a_1 =$	0.096954
$a_2 =$	-0.034164
$a_3 =$	-0.0069427
$a_4 =$	-0.00093078

Thus, the interval for each 20 bits is $(-8, 8)$. Table 5 shows the final weights for Trial F. Figure 7 shows the evolution of the population over several generations for Trial F.

The results are revealing. For very small populations, sized from 100 to 1000 genes, the GAFE is unable to isolate a search area with find a suitable set of weights; that is, it gets "stuck" on a local optimum. It is clear that larger populations fare better than smaller populations, regardless of the number of generations evolved. Even in the final experiment, where there are 10.000 genes in the pool, the final solution was found by chance, due, most likely, to the migration operator. It is unclear that an optimal solution would be converged upon, even if the gene pool is allowed to evolve for many generations. This may be attributed to any number of factors, such as the actual mathematical structure of $\widehat{FE}$, the parameterization of the weights, or the choice of genetic operators and their probabilities. However, it is not difficult to see that the the weights evolved in Trial F produce a usable, if not noisy, image. This noise can be easily removed by a simple thresholding technique.

The mathematical structure of GAFE poses a unique problem. There is no generalized form of fractal error as it is defined above. A generalized form would make it possible to generate a type of convolution mask which could be used to compute fractal error in a timely fashion. Forcing a mathematical structure to find the approximation is restrictive. In the future, it may be prudent to investigate the evolution of a mathematical expression using a GP. This would expand the search space from merely searching the weights for $\widehat{FE}$ to searching the infinite space of reusable functions to find the approximation.

The approximation was tested on several aerial images, include a SAR image. The results for the "Washington D.C." and "Alameda" test images are shown in Figures 8 and 9. The results for the SAR image are shown in Figure 10. Table 6 shows the RMS error for the approximation relative to the fractal error algorithm. Table 7 shows the computation times of the fractal error algorithm and the GAFE for images of several different sizes.

Table 6: RMS Errors Of Test IMages With Respect To Original FE Images

Image	Normalized RMS Errors
Alameda	0.109
Washington. DC	0.099
SAR	0.114

(a) Trial A (b) Trial B (c) Trial C
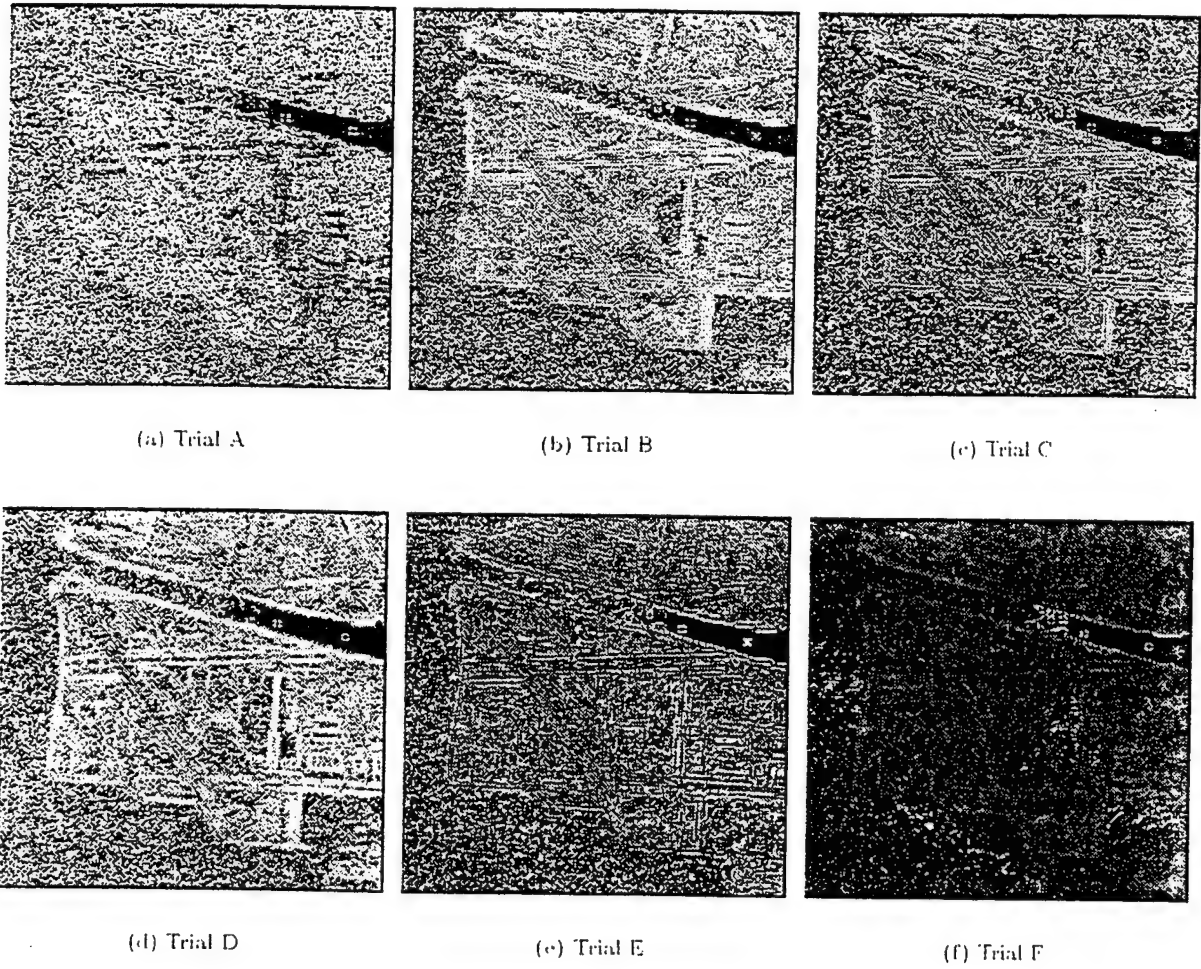
(d) Trial D (e) Trial E (f) Trial F

Figure 6: Resulting $\widehat{FE}$ images from each GAFE trial described in Table 4.

Table 7: Computation Time For The Original FE Algorithm and The GAFE

Size	FE Algorithm (seconds)	GAFE (seconds)
256x256	1.43	0.43
227x425	2.14	0.64
589x409	5.59	1.65
512x486	5.68	1.69
974x723	16.36	4.85
1267x941	27.71	8.22
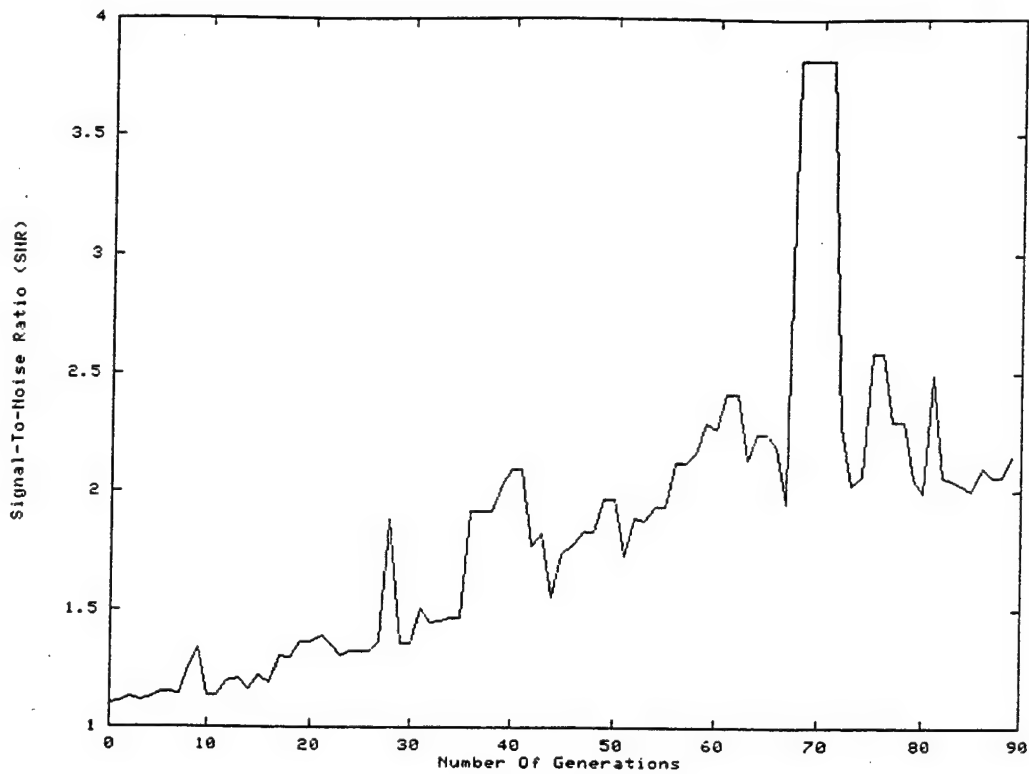
Figure 7: Plot of Trial F with SNR vs. number of generations.

6   Conclusions

A genetic approximation to fractal error has been developed in this paper. The approximation provides usable results. The GAFE approximation provides significant reductions in computation time. A faster approximation may be possible if the calculation of the natural log function is implemented as a lookup table. The GAFE may also be parallelized, splitting the image up into subimages, passing the approximation's mathematical structure and the subimages off to as many processors as are available. This may also provide a result in a more timely fashion. Finally, a genetic programming approach is being explored to evolve a mathematical structure to more accurately approximate fractal error.
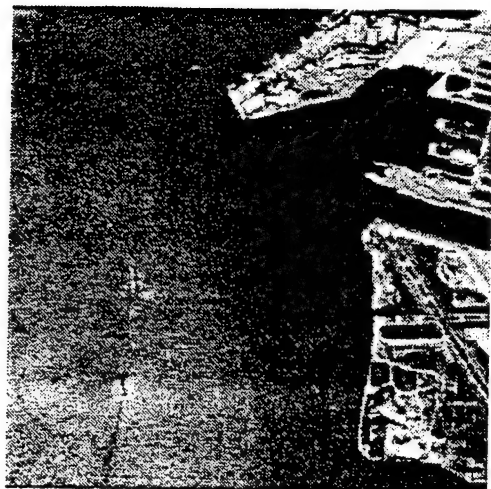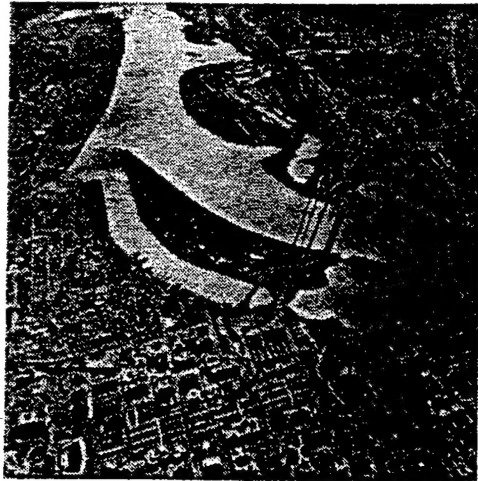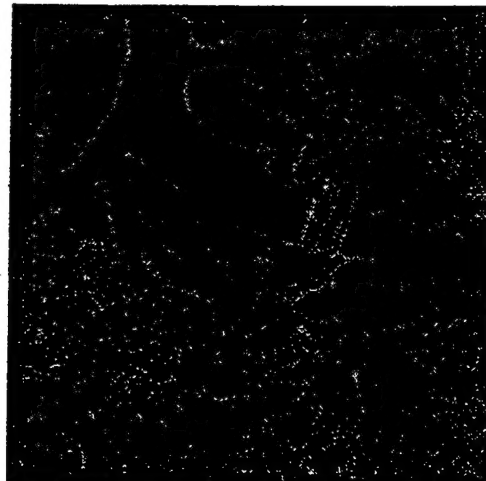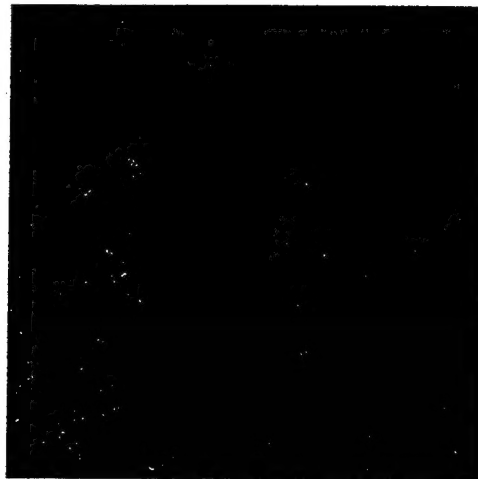
(a) Original

(b) FE Algorithm
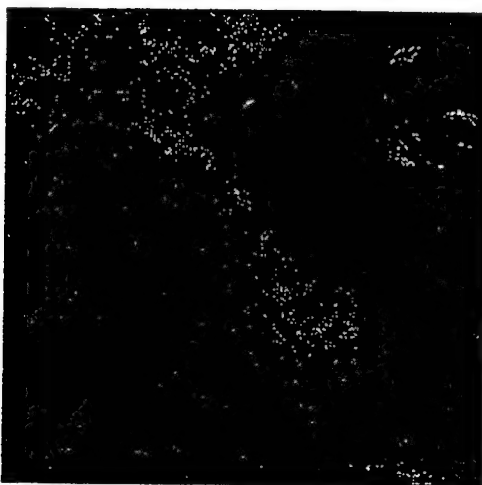
(c) GAFE

Figure 8: (a) The original "Alameda" image (256x256), the outputs given by the (b) fractal error algorithm, and (c) the genetic approximation.
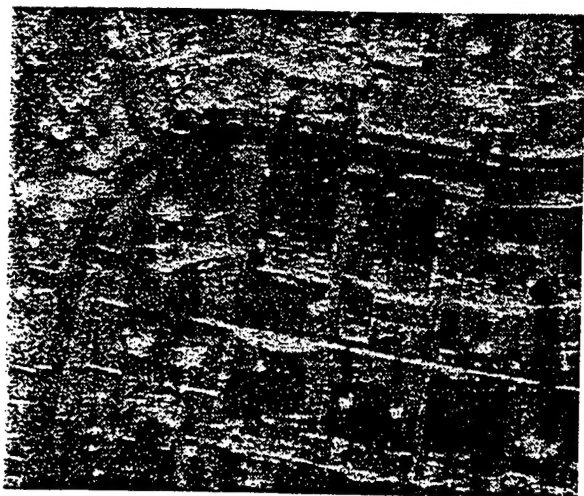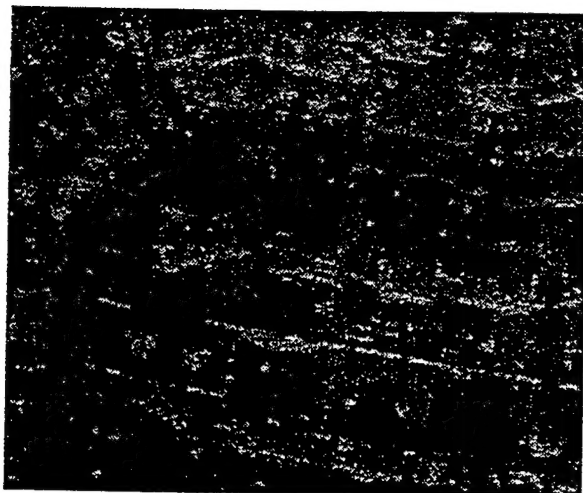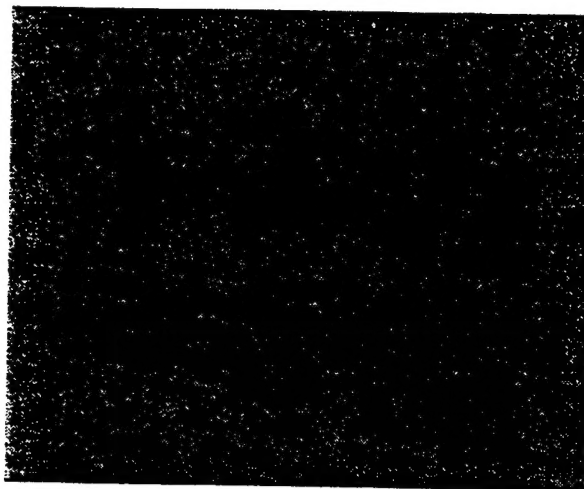
(a) Original

(b) FE Algorithm

(c) GAFE

Figure 9: (a) The original "Washington D.C." image (512x480). the outputs given by the (b) fractal error algorithm. and (c) the genetic approximation.

(a) Original

(b) FE Algorithm

(c) GAFE

Figure 10: (a) The original SAR image (512x480), the outputs given by the (b) fractal error algorithm, and (c) the genetic approximation.

References

[1] B. E. Cooper, D. L. Chenoweth, and J. E. Selvage. "Fractal error for detecting man-made features in aerial images", *Electronics Letters*, vol. 30, no. 7, pp. 554–555, 1994.

[2] E. D. Jansing, D. L. Chenoweth, and J. Knecht, "Feature detection in synthetic aperture radar images using fractal error", in *Proceedings of the IEEE Aerospace Conference*, 1997, vol. 1, pp. 187–195.

[3] E. D. Jansing, B. S. Allen, and D. L. Chenoweth, "Edge enhancement using the fractal error metric", in *Proceedings of the 1st International Conference on Engineering Design and Automation*, 1997.

[4] A. P. Pentland, "Fractal-based description of natural scenes", *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. PAMI-6, no. 6, pp. 661–674, 1984.

[5] J. M. Keller, S. Chen, and R. M. Crownover, "Texture description and segmentation through fractal geometry", *Computer Vision, Graphics and Image Processing*, vol. 45, pp. 150–166, 1989.

[6] S. Peleg, J. Naor, R. Hartley, and D. Avnir, "Multiple.resolution texture analysis and classification", *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. PAMI-6, no. 4, pp. 518–523, 1984.

[7] C. V. Stewart, B. Moghaddam, K. J. Hintz, and L. M. Novak, "Fractional brownian motion models for synthetic aperture radar imagery scene segmentation", *Proceedings of the IEEE*, vol. 81, no. 10, pp. 1511–1521, October 1993.

[8] B. Moghaddam, K. J. Hintz, and C. V. Stewart, "Fractal image compression and texture analysis", in *SPIE 19th AIPR Workshop*, October 1990.

[9] B. Moghaddam, K. J. Hintz, and C. V. Stewart, "Dimension and lacunarity measurements of IR images usning hilbert scanning", in *SPIE International Symp. on Optical Engineering and Photonics in Aerospace Sensing*, April 1991.

[10] B. E. Cooper, *Fractional Brownian Motion For Representation Of Natural Image Texture*, PhD thesis, University of Louisville, Louisville, KY, 1994.

[11] M. C. Stein, "Fractal image models and object detection", in *Proceedings of the SPIE: Visual Communications and Image Processing II*, 1987, vol. 845.

[12] J. L. Solka, C. E. Priebe, and G. W. Rogers, "An initial assessment of discriminant surface complexity for power law features", *Simulation*, vol. 52, no. 5, 1992.

[13] G. W. Rogers, J. L. Solka, and C. E. Priebe, "A PDP approach to localized fractal dimension computation with segmentation boundaries", *Simulation*, vol. 65, no. 1, 1995.

[14] M. La Brecque, "Fractal applications", *Mosaic*, vol. 17, no. 4, pp. 34–48, Winter 1986/1987.

[15] K. Falconer, *Fractal Geometry-Mathematical Foundations and Applications*, John Wiley and Sons, Ltd., New York, NY, 1990.

[16] B. B. Mandelbrot, *The Fractal Geometry of Nature*, Freeman, San Francisco, CA, 1983.

[17] H.-O. Peitgen, H. Jurgens, and D. Saupe, *Chaos and Fractals: New Frontiers of Science*, Springer-Verlag, New York, 1992.

[18] G. A. Edgar, Ed., *Classics On Fractals*, Addison-Wesley, Reading, MA, 1993.

[19] B. H. Kaye, *A Random Walk Through Fractal Dimensions*, VCH, New York, 1989.

[20] P. Brodatz, *A Photographic Album For Artists and Designers*, Dover, New York, 1966.

[21] B. B. Mandelbrot, *Fractals: Form, Chance and Dimension*, W. H. Freeman and Co., San Francisco, 1977.

[22] L. J. Bain and M. Engelhardt, *Introduction to Probability and Mathematical Statistics*. PWS-Kent, Boston. 1992.

[23] J. C. Potts, T. D. Giddens, and S. B. Yadav, "The development and evaluation of an improved genetic algorithm based on migration and artificial selection", *IEEE Transactions on Systems. Man and Cybernetics*, vol. 24, no. 1. January 1994.